KOCAELİ ÜNİVERSİTESİ * FEN BİLİMLERİ ENSTİTÜSÜ

H.264/AVC'DE HIZLI HAREKET KESTİRİMİ İÇİN DÜŞÜK GÜÇLÜ DONANIM MİMARİLERİ VE ALGORİTMA TASARIMI

DOKTORA TEZİ

Elektronik ve Haberleşme Yük. Müh. Anıl ÇELEBİ

Anabilim Dalı: Elektronik ve Haberleşme Mühendisliği

Danışman: Prof. Dr. Sarp ERTÜRK

KOCAELİ, 2008

ÖNSÖZ VE TEŞEKKÜR

Bu tez çalışması, 2008 yılında kısa adı KULIS olan Kocaeli Üniversitesi İşaret ve Görüntü İşleme Laboratuarında çalışmalarına başlanılan TÜBİTAK ile Kore Araştırma Vakfı işbirliği programı kapsamında, 107E179 no'lu proje olarak desteklenen "Düşük Karmaşıklığa Sahip Hareket Kestirim Yaklaşımları ve Bunların Yonga Üzeri Sistem Tasarımları" isimli proje kapsamında gerçekleştirilen çalışmalardan bazılarını içermektedir. Bu çalışmada önerdiğim donanım mimarilerinin ve hareket kestirimi yaklaşımlarının bu alanda ileride yapılacak çalışmalar için iyi bir referans çalışma olmasını ümit ediyorum. Özellikle, önerdiğim donanım mimarileri, bilgisayar benzetiminden çok gerçek hayatta çalışabilecek şekilde tasarlamaya gayret ettim. Sunduğum sonuçlarda özellikle tasarımların gerçek hayatta gösterecekleri performansları ile ilgili verilere ağırlık verdik. Umarım yaptığımız çalışmaların video kodlama alanına küçük de olsa bir katkısı olur.

Tez izleme komitemdeki öğretim üyesi Yrd. Doç Dr. M. Kemal Güllü'ye tez çalışmalarım sırasında düşüncelerini benimle paylaştığı için teşekkür ederim. Prof. Dr. Günhan DÜNDAR'a gerek doktora ders döneminde gerekse tez çalışmalarım sırasında verdiği destek dolayısıyla çok teşekkür ederim. Tez çalışmalarım sırasında tanıştığım ve sonrasında gerek yüz yüze gerekse uzun telefon görüşmelerinde fikirlerini benimle paylaşan sevgili hocam Yrd. Doç. İlker HAMZAOĞLU'na verdiği destekten ötürü teşekkür ederim. Lisans ve yüksek lisans çalışmalarım sırasında bana çeşitli araştırma projelerinde çalışma olanağı sunan Yrd. Doç. Dr. Ali TANGEL'e, bir araştırmacı olarak bugünlere gelmemde harcadığı emeklerinden ötürü teşekkür ederim. Tez danışmanım Prof. Dr. Sarp ERTÜRK ile tanışmam, lisans eğitimimde olmuştu ama bir türlü, çalışma alanımı onun uzmanlık alanıyla kesiştirme fırsatını bulamamıştım. Şimdi bu fırsatı yakaladığım için duyduğum mutluluk tarif edilemez. O olmasaydı bu tez çalışmasını tamamlamam mümkün olmazdı. Kendisine bana verdiği sınırsız destekten ötürü çok teşekkür ederim.

Sevgili arkadaşım, Yrd. Doç Dr. Oğuzhan URHAN'a her şeyden önce benim arkadaşım olduğu için çok teşekkür ederim. Bunun dışında, video kodlama konusunda bilgi birikimini benimle paylaştığı, tezin organizasyonuna yaptığı katkı ve yaşadığım diğer birçok teknik sıkıntıda bana sıkılmadan yardım ettiği; bu tezi yazarken, alıp başımı gitmeden belirli sınırlar içinde kalmamı sağladığı; yazdığım teknik makaleleri herkesten önce ve severek düzelttiği için kendisine sonsuz teşekkür ediyorum.

Güzel ve samimi bir çalışma ortamını paylaştığım KULİS'deki tüm araştırmacı arkadaşlarıma, birlikte paylaştığımız ve gelecekte de paylaşmaya devam edeceğimiz güzel zamanlar için teşekkür ediyorum.

Aileme, bugüne gelmemde verdikleri bütün emekler ve her zaman onları arkamda hissetmemi sağladıkları için çok teşekkür ederim. Umarım bu çalışma ile verdikleri emeklerin karşılıklarını bir parça da olsa onlara geri ödeyebilmişimdir.

Sevgili ve biricik Aysun’a, bana bir eş olarak verdiği sonsuz desteğin dışında, bir meslektaş olarak, bu tezi yazarken yanımda olması, yapıcı eleştirileri, düzeltmeleri ve hiç esirgemediği moral desteğinden ötürü sonsuz teşekkür ediyorum.

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

SİMGELER DİZİNİ

Simgeler

N	:	Blok boyutu indisi
p	:	Arama aralığı indisi
$c(i,j)$	:	*i,j* konumundaki güncel blok pikseli
$s(i,j)$	:	*i,j* konumundaki arama penceresi pikseli
$MFT(m,n)$	:	*m,n* konumundaki makro-bloğun mutlak farklar toplamı değeri
$1/K$	:	Hareket kestirimi arama aralığı alt örnekleme oranı
$\bullet$	:	Mantıksal Ve
$\|\|$	:	Mantıksal VEYA
$\oplus$	:	EXOR

Kısaltmalar

1BD	:	1-Bit Dönüşümü
2BD	:	2-bit Dönüşümü
AEY	:	Ardışık eleme yöntemi
ASIC	:	Uygulamaya Özel Tümleşik Devre
AVC	:	Advanced Video Coding
BÇ	:	Bağımlılık çizelgeleri
BDU	:	Bit Düzlemi Uyumlama
BU	:	Blok Uyumlama
CIF	:	Common Intermediate Format
CLB	:	Configurable Logic Block
CM	:	İlişki ölçüsü
CNNMP	:	Constraint number of non-matching points
Ç1BD	:	Çarpmasız 1-Bit dönüşümü
ÇAK	:	Çerçeve Arası Kodlama
ÇAÖK	:	Çerçeve Arası Öngörücü Kodlama
ÇBÖT	:	Çok büyük ölçekli tümleştirme
ÇİK	:	Çerçeve İçi Kodlama
ÇSAEY	:	Çok seviyeli ardışık eleme yöntemi
DBBHK	:	Değişken blok boyutlu HK
DUK	:	Değişken Uzunluklu Kodlama
EDGE	:	Enhanced Data rates for GSM Evolution
FDKK	:	Farksal Darbe Kodlu Kiplenim
FPGA	:	Alan Programlanabilir Kapı Dizileri
GSM	:	Global System for Mobile communications
HBUY	:	Hızlı blok uyumlama yöntemi
HK	:	Hareket Kestirimi
HV	:	Hareket Vektörü
HVTDD	:	Hareket vektörü temelli doğrusal dizi
ITU	:	International Telecommunication Union
İP	:	İşlem Parçası

JPEG	:	Joint Photograpghic Experts Group
K-1BD	:	Kısıtlanmış 1-Bit dönüşümü
KBEY	:	Kısmi bozulma eleme yöntemi
KBEY	:	Kısmi bozulma elemesi yöntemi
KGKBDU	:	Kesik Gray kodlanmış BDU
KHK	:	Kesirli Hareket Kestirimi
KPTDD	:	Kaynak piksel temelli doğrusal dizi
KPTDD	:	Kaynak Piksel Temelli Doğrusal Diziler
KS	:	Kaydırmalı saklayıcı
KUNS	:	Kısıtlanmış Uyumsuz Nokta Sayısı
LUT	:	İşlem okuma tablosu
MB	:	Makro-blok
MFT	:	Mutlak Farklar Toplamı
MMD	:	Max-Min Difference
MPEG	:	Moving Pictures Experts Group
NNMP	:	Number of non-matching points
NTB	:	Number of Truncated Bits (Kesilen bit sayısı)
OMF	:	Ortalama mutlak fark
OMH	:	Ortam mutlak hata
PFS	:	Piksel farkı sınıflandırma
PSNR	:	En büyük işaret gürültü oranı
RISC	:	Reduced Instruction Set Computer
SAS	:	Saklayıcı Aktarımı Seviyesi
SBBHK	:	Sabit blok boyutlu HK
TABUY	:	Tam arama temelli blok uyumlama yöntemi
THK	:	Tamsayı Hareket Kestirimi
UNS	:	Uyumsuz Nokta Sayısı
VYK	:	Verilerin yeniden kullanımı
XOR	:	Mantıksal özel veya işlemi
YFBU	:	Yüksek frekans bileşeni uyulmaması
YÜ	:	Yonga üzeri

H.264/AVC'DE HIZLI HAREKET KESTİRİMİ İÇİN DÜŞÜK GÜÇLÜ DONANIM MİMARİLERİ VE ALGORİTMA TASARIMI

Anıl ÇELEBİ

Anahtar Kelimeler: Video Kodlama, Hareket Kestirimi,1-Bit Dönüşümü, Sistolik Diziler, FPGA, ASIC, SOC, Hareket Kestirimi Mimarisi, Doğrusal Diziler, Paralel Mimariler

Özet: Bu tez çalışmasında H.264/AVC kodlayıcılarda hareket kestirimi (HK) işlemini yapmak üzere düşük bit gösterimi temelli yeni HK yöntemleri ve bu yöntemlerin yonga üzeri sistem (SoC) şeklinde veya uygulamaya özel tümdevre tasarımı (ASIC) şeklinde gerçekleştirilebilmelerini sağlayacak yeni mimariler önerilmiştir. Yakın geçmişte önerilmiş olan iki farklı düşük bit gösterimi temelli HK yöntemi öncelikle hareket vektörü temelli doğrusal dizilerle (HVTDD) gerçeklenmiştir. Ayrıca bu yöntemler için ilk kez kaynak piksel temelli doğrusal dizi (KPTDD) yapısını kullanan bir donanım mimarisi geliştirilmiştir. KPTDD mimarisinin bu tez kapsamında düşük bit gösterimine sahip HK yöntemlerine uygulanması ile HVTDD mimarisine göre güç tüketimi 2 kattan fazla azaltılmıştır.

Bu tez kapsamında ayrıca, bit kesme tekniği temelli yeni bir HK yöntemi ve bu yöntemin donanım mimarisi geliştirilmiştir. Önerilen yöntemde doğrudan piksellerin ikili karşılıkları yerine Gray kodu karşılıklarının belli sayıdaki en değerliksiz bitlerinin kesilmesi ile HK doğruluğu önemli ölçüde geliştirilmiştir. Bu yöntem, literatürdeki diğer düşük bit gösterimi temelli yöntemlere göre HK doğruluğu bakımından en iyi sonucu vermektedir.

Düşük bit gösterimi temelli HK yaklaşımlarının kesirli hareket kestirimine uygulanması literatürde detaylı incelenmemiş bir konudur. Bu tez kapsamında tamamı ile düşük bit gösterimi kullanılan özgün bir kesirli hareket kestirim yöntemi geliştirilmiş ve bu yönteme özgü donanım mimarisi tasarlanmıştır. Deneysel sonuçlar, tamamı ile düşük bit gösteriminde yapılan kesirli HK işleminin hareket kestirim başarımını artırdığını göstermiştir.

Önerilen donanım mimarileri, genel amaçlı bir FPGA yongası üzerinde gerçeklenmiştir. Tasarlanan mimarilerin, HK performansı, güç tüketimi, yonga alanı, çalışma frekansı gibi özellikleri geçmişte önerilen diğer donanım mimarileri ile karşılaştırılmış ve geliştirilen yöntemlerin üstün bir başarım sağladığı deneysel sonuçlarla gösterilmiştir.

LOW POWER HARDWARE ARCHITECTURES AND ALGORITHM DESIGN FOR FAST MOTION ESTIMATION FOR H.264/AVC

Anıl ÇELEBİ

Keywords: Video Coding, Motion Estimation,1-Bit Transform, Systolic Arrays, FPGA, ASIC, SOC, Motion Estimation Architecture, Linear Arrays, Parallel Architectures

Abstract: In this dissertation, new low bit depth representation based motion estimation (ME) approaches and their novel hardware architecture is proposed for H.264/AVC encoders. Initially, two of the recently proposed low bit depth representation based ME approaches are implemented based on motion vector based linear array (MVBLA) architecture. Furthermore, the hardware implementations of these approaches are carried out by making use of the source pixel based linear array architecture (SPBLA) in this work, which is first in the literature. The power consumption of low bit depth representation based ME approaches is reduced more than two times by utilizing the SPBLA architecture in the scope of this dissertation.

A novel bit-truncation technique based ME approach and its hardware architecture is developed in the scope of this dissertation as well. ME accuracy is significantly improved by truncating a certain number of least significant bits of the Gray coded pixels instead of the original pixels directly.

In the scope of this dissertation a novel all binary low bit depth representation based sub-pixel ME approach is developed and its hardware architecture is designed. Experimental results have shown that ME performance is improved by the all binary low bit depth representation based fractional pixel ME.

The proposed hardware architectures are implemented on a general purpose FPGA chip. Furthermore, the proposed architectures are compared to existing hardware architectures in terms of the ME performance, power consumption, chip area, and operating frequency and it is shown through experimental results that the developed approaches have superior performance.

1. GİRİŞ

1.1 Nereden, Nereye?

Gelişen teknoloji ile birlikte ortaya çıkan çok çeşitli, görsel, işitsel uygulamalar için gereksinim duyulan farklı işlemlerin analog bilgiler üzerinde gerçekleştirilebilmesinin önünde sayısız engeller bulunmaktadır. Bu nedenle çok uzun yıllar önce bilginin 1 ve 0'lardan oluşan ayrık ve sayısal işaretler ifade edilmesinin yolları aranmaya başlanmıştır. İlk başlarda ihtiyaç duyulan uygulama çeşitliliği sınırlı olduğundan analog sistemler uzunca bir süre cazibesini korumuştur. Bu uzun zaman yayılan geçiş sürecinde aşılan bazı temel zorluklar bulunmaktadır. Bunlardan bir tanesi analog olarak sonsuz değer alabilecek bir işaretin sayısal olarak ifade edilmesidir. Bunun yanında analog işaretlerin sonlu bir çözünürlükte ifade edilmesi dışında bu sonlu çözünürlüğün asgari bir anlaşılabilirlik seviyesini de sağlaması zorunluluğu vardır.

Teknolojinin geldiği noktada bizi sınırlayan çeşitli değişkenler vardır. Bunlar uygulama alanlarına göre farklılık göstermektedir. Örneğin haberleşme sistemleri için bu sınırlama iletim bant genişliği iken sayısal bir video kamera için sistemin sahip olduğu sınırlı bellek miktarı ve işlem kapasitesidir. Bu gibi sınırlamaların ortadan kaldırılması için tercih edilen yöntemlerden birisi sayısal etkin bir şekilde işlemek ve saklamaktır. Sayısal verinin daha etkin saklanması ve iletilmesini sağlamak için en sık başvurulan yöntemlerden bir tanesi verilerin sıkıştırılmasıdır.

Tek boyutlu işaretlerin saklanması ve iletilmesi günümüz teknolojisinin geldiği noktada çok büyük bir sıkıntı yaratmasa da video gibi üç boyutlu ve çok fazla miktarda bilgi içeren bir verinin iletimi ve saklanması halen üzerinde çalışılan önemli bir araştırma konusudur. Bu amaçla sayısal hale getirilmiş video verisinin etkin bir biçimde kodlanması yakın geçmişte üzerinde çok yoğun çalışma yapılmış bir alandır. Örneğin, yeni nesil haberleşme sistemlerinde, sınırlı bant genişliğine sahip iletim kanalları üzerinden video verisinin gerçek zamanlı aktarımı önemli bir yeniliktir. Ayrıca günümüzde internet üzerinde akan verinin önemli bir kısmını da video oluşturmaktadır. Diğer taraftan televizyon yayıncılığının da çok kısa bir zaman

içerisinde tamamen sayısal ortamda gerçekleştirileceği göz önünde bulundurulduğunda etkin video kodlama yöntem ve donanımlarının geliştirilmesi büyük bir öneme sahiptir.

1.2 Video Kodlama

Sayısal video, en genel haliyle belirli ayrık zaman dilimlerinde bulunan iki boyutlu görüntü çerçevelerinin bir araya gelmesi ile oluşmuş veri olarak tanımlanabilir. Görüntü çerçeveleri ise piksel denilen görüntü elemanlarından oluşmaktadır. Görüntü çerçevelerinde görüntü elemanları dikdörtgensel bir matris şeklinde yer almaktadır. Siyah beyaz video, ayrık görüntü çerçevelerine ilişkin ışıklılık verilerini içermektedir. Siyah beyaz sayısal video dizilerinde her bir piksel genellikle 8 bit ile temsil edilir dolayısıyla görüntü 256 ayrık seviyede nicemlenir. Renkli video dizilerinde ise birden fazla (tipik olarak üç) renk uzayı vardır. Örneğin RGB denilen, kırmızı, yeşil ve mavi renklerinin farklı oranlarda birleşiminden oluşan video dizilerinde veri boyutu siyah beyaz görüntülere göre üç kat daha fazladır çünkü her bir ana rengin bir piksel içerisindeki oranını gösteren üç farklı matris bulunmaktadır. Bu üç matris, kendi içerisinde 8 bit ile gösterilen değerlere sahiptir ve gözle görülen renkli pikseller her bir ana rengin belli oranda karışımından oluşur.

Görüldüğü gibi sayısal video verisinde siyah beyaz görüntüden renkli görüntüye geçildiğinde veri miktarı da artmaktadır. Basit bir hesap yapılacak olursa RGB renk uzayında gösterilen, 640×480 boyutundaki ve 30 $çerçeve/saniye$ hızındaki bir video çerçevesinin gerektirdiği bellek boyutu (640×480(piksel)) × 30(Çerçeve hızı) × 8bit(Piksel derinliği) × 3(RGB) = 27.648 $MByte$'dır. Dolayısıyla video verisinin, haberleşme sistemlerindeki sınırlı bant genişliği ve kayıt ortamlarının sınırlı kapasiteleri dikkate alındığında sıkıştırılmadan iletilmesi veya saklanması mümkün değildir.

Video işaretlerinin istatistiksel olarak incelenmesi ardışık video çerçeveleri arasında yoğun bir ilişki olduğunu göstermiştir. Ayrıca video çerçeveleri içerisindeki piksellerin kendi aralarında da yoğun bir ilişki vardır. Ek olarak, insan gözü bazı uzamsal ve zamansal artıklıklara karşı duyarsızdır. Bahsedilen bu noktalar göz önünde bulundurulursa daha yüksek oranlarda sıkıştırma başarımı elde edilebilir. Sonuç olarak görsel kaliteyi kabul edilebilir seviyelerde tutarak kayıplı sıkıştırma yöntemleri ile videonun bit oranı (bir rate) azaltılabilir [1].

Video çerçevelerini kendi içerisinde sıkıştırmak için sadece uzamsal ilişki kullanılabilir. Böyle bir kodlama çerçeve-içi kodlama (ÇİK – intra frame coding) olarak adlandırılır ve örneğin imge kodlama için geliştirilmiş JPEG (Joint Photographic Experts Group) kodlamanın temelini oluşturur. Bununla beraber videoda zamansal ilişkiden de faydalanılırsa kodlama verimliliği artırılabilir. Bu tür kodlamaya çerçeveler-arası kodlama (ÇAK – inter frame coding) denilmektedir. Çerçeveler-arası öngörücü kodlama (ÇAÖK) (IFPC-Inter frame predictive coding) yöntemi, ISO MPEG serisi [2-4] ve ITU-T H.26X serisi video kodlama standartlarında kullanılmaktadır [5-7].

Yukarıda bahsedilen kapsam doğrultusunda video kodlama işleminin temel olarak aşağıda verilen üç farklı artıklık giderimi yaklaşımı üzerine kurulduğu söylenebilir.

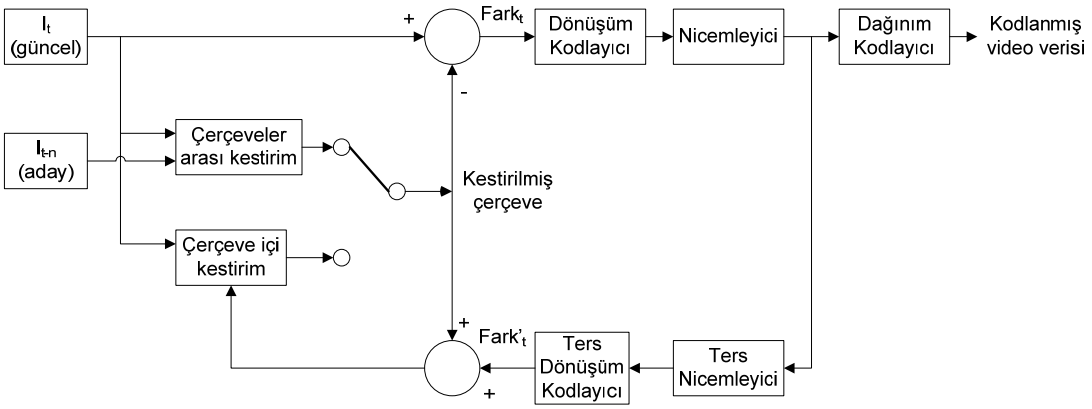
1. Uzamsal artıklık giderimi: Çerçeve-içi kestirim (intra prediction) ve/veya dönüşüm kodlaması (Transform Coding) gibi yöntemler kullanarak imge içerisindeki pikseler arasındaki uzamsal artıklık giderilmeye çalışılır.

2. Zamansal artıklık giderimi: Ardışık video çerçeveleri arasındaki benzerliğin kestirilmesiyle elde edilen farkın kodlanması yoluyla video çerçeveleri arasındaki artık bilgi için fazladan veri kodlama engellenmeye çalışılır.

3. Dağınım (Entropy) kodlaması: Bu işlemde, sıkıştırılmış veri içerisindeki istatistiksel artıklığı gidermek amacıyla yaygın olarak, kayıpsız değişken uzunlukta kodlama (DUK) (VLC-Variable Length Coding) yöntemleri kullanılır.

Yukarıda açıklanan üç temel artıklığı göz önüne alacak şekilde geçmişte geliştirilen video kodlama yaklaşımları temel olarak Şekil 1.1'de gösterilen yapıda oluşturulmuştur. Verilen kodlayıcı yapısında görüldüğü gibi, kodlanacak imge çerçevesi, ya çerçeve içi ya da çerçeveler arası kodlama yaklaşımı ile öncelikle kestirilmektedir. Çerçeve içi kestirilen çerçeveler, I-çerçevesi (Intra frame) olarak adlandırılırken, çerçeveler arası kestirilen çerçeveler, kullanılan referans çerçevenin kestirilme durumuna göre P-çerçevesi (pretictedframes) veya B-çerçevesi (bi-directionally predicted frame) adını almaktadır. Sonrasında elde edilen kestirilmiş imge çerçevesi ile kodlanacak güncel çerçevenin farkı dönüşüm kodlayıcısına girmektedir. Elde edilen dönüşüm katsayıları, video kalitesini veya istenen bit miktarını ayarlamak için nicemlenmektedir. Son olarak, nicemlenmiş dönüşüm

katsayılarındaki istatistiksel artıklığı gidermek için dağınım kodlaması yaklaşımlarından faydalanılmaktadır.



Şekil 1.1: Tipik bir video kodlayıcı

Bütün video kodlama standartları, temel olarak Şekil 1.1'de verilen kodlayıcı yapısını kullanırken, gösterilen yapıdaki her öbeğin çalışma şeklinde gereksinimler doğrultusunda değişiklikler yaparak video kodlama başarımını artırmaya çalışmaktadır. Örneğin H.264/AVC'den önceki video kodlama standartlarında hareket kestirimi (HK) için tek bir aday çerçeve kullanılırken, H.264/AVC'de çoklu aday çerçeve kullanımı ile HK başarımı artırılmaya çalışılmıştır. Yine H.264/AVC'de daha gelişmiş çerçeve içi kestirim yaklaşımları kullanılarak kodlama verimliliğini artırma yolları aranmıştır. Ayrıca, nicemlenmiş dönüşüm katsayılarının kayıpsız kodlanması için daha etkin çalışan yöntemler H.264/AVC'de benimsenmiştir. Ancak video kodlama başarımını artıran bu yöntemlerin hepsi ek bir hesap yükünü de beraberinde getirmektedir. Video kodlama ile ilgili daha kapsamlı bilgiler için [8]'den faydalanılabilir.

Yukarıda açıklanan artıklık giderimi yöntemlerinden, video kodlama miktarı üzerinde en fazla katkıya sahip olan kısım çerçeveler arası artıklığın giderimidir. Çerçeveler arası artıklık, tipik olarak video çerçeveleri belli boyutlarda örtüşmeyen bloklara bölünerek bu bloklar üzerinde gerçekleştirilir. Bütün video kodlama standartlarında kodlayıcının işlem yükünün en büyük kısmı HK'den kaynaklanmaktadır. Bu nedenle ve tezin ana kapsamını oluşturduğundan HK, takip eden alt bölümde daha detaylı olarak ele alınacaktır.

1.3 Hareket Kestirimi

Çerçeveler arası kestirim işleminin ardındaki temel mantık, önceden kodlanmış bir video çerçevesini kullanarak hali hazırda kodlanmakta olan çerçeveyi olabildiğince az veri kullanarak kodlamaktadır. Bu amaçla kodlanmış çerçeve ile kodlanacak olan güncel çerçevenin farkını alıp bu farkı kodlamak temel bir yaklaşım olarak ele alınabilir. Bu yaklaşım video çerçevelerindeki arka plan gibi durağan bölgelerin verimli şekilde kodlanmasına olanak sağlarken, hareketli bölgelerin kodlanması sırasında referans olarak kullanılan kodlanmış çerçeveye göre önemli miktarda fark bilgisi oluşacaktır. HK işlemi bu noktada devreye girerek, kodlanacak güncel çerçevenin daha etkin kodlamasına olanak sağlar. Bunun için tipik olarak kodlanacak güncel video çerçevesi örtüşmeyen bloklara ayrılır. Sonrasında, her bir blok için zaten kodlanmış bulunan aday çerçevedeki belirli bir arama bölgesinde arama yapılarak aranan bloğa en uyumlu blok bulunup bu blok, kestirim çerçevesinde uygun yere konur. Bu işlem, kodlanacak güncel video çerçevesindeki her bir blok için yapılarak kestirim çerçevesi oluşturulur. Bu yolla, kodlanmış çerçeve yerine, elde edilen kestirim çerçevesi kullanılarak yapılan fark alma işlemiyle, kodlanması gereken veri miktarı büyük ölçüde azaltılmaktadır. Böylelikle kodlama verimliliği arttırılmaktadır.

HK, işlem yükü çok fazla olan bir süreçtir; bu süreç kodlayıcıdan kodlayıcıya değişmekle birlikte, güncel video kodlama standartlarından H.264/AVC'de toplam işlem yükünün %70'inin üzerindedir [9]. Ayrıca, günümüzde en güncel video kodlama standardı olan H.264/AVC'de, daha iyi bir kestirim sağlamak üzere HK işleminin birden fazla referans çerçevesinde yapılması önerilmektedir [7]. HK işleminin kodlayıcıya getirdiği yükün artmasıyla birlikte gerçek-zamanlı uygulamalar için daha düşük işlem yüküne sahip ve hızlı çalışan HK algoritmalarının kullanılması kaçınılmaz olmuştur. Çevrim dışı (offline) uygulamalar için HK işleminin getirdiği yük çok fazla bir önem taşımamakta ve yazılım tabanlı çözümler kullanılabilmektedir.

Mobil uygulamalar gibi; gerçek zamanlı çalışma gerektiren, zaman ve güç kısıtlaması bulunan uygulamalarda yazılımsal video kodlama çözümlerinin kullanılması genellikle mümkün olmamaktadır. Bu nedenle yüksek işlem kapasitesine sahip donanımsal çözümler üzerine yakın geçmişte başlayan bilimsel çalışmalar hali hazırda çok yoğun bir şekilde devam etmektedir.

Video kodlayıcılar tamamen donanımsal olarak tasarımlanabileceği gibi sadece işlem yükü yüksek olan kısımların donanımsal olarak gerçeklenmesi de yaygın olarak tercih edilen bir yaklaşımdır. Bu kapsamda sayısal işaret işleyicilerden (digital signal processor-DSP), alan programlanabilir kapı dizileri (field programmable gate arrays-FPGA) ve uygulamaya özgü tümleşik devre (application spesific integrated circuit-ASIC) tasarımına kadar geniş bir çerçevede farklı çözümler önerilmiştir.

DSP tabanlı çözümler, doğrudan yazılımsal çözümlere göre kodlama performansında iyileştirme ve geliştirme aşamasında esneklikler sağlamasına karşın, gerçek zamanlı çalışabilme, güç tüketimi ve yüksek miktarda veri işleme kapasitesi bakımından yetersiz kalabilmektedir. Öte yandan ASIC tasarım, yüksek işlem kapasitesi, güç tüketimi ve gerçek zamanlı çalışma noktalarında en iyi çözümü sunuyor olsa da gerek ilk örnekleme maliyeti, gerekse tasarım sürecinin uzunluğu ve tasarımdaki değişikliklerin oldukça zahmetli olması bu yaklaşımın önemli dezavantajlarındandır. Bu noktada, yapılan tasarımların kolayca gerçeklenebilmesi, yeniden düzenlenebilirliğe açık olması ve üretim açısından ilk örnekleme maliyetini ortadan kaldırdığı için FPGA tabanlı donanımsal çözümlerden yararlanmak yaygın şekilde tercih edilen bir yol olmuştur.

HK işlemdeki aşamaların düzenli bir yapıda olması, bu işlemlerin kolayca paralelleştirilmesine olanak sağlamaktadır. Sistolik dizi tabanlı mimariler geçmişte bu amaçla yaygın ve etkin olarak kullanılmıştır. Sistolik diziler, belirli bir işi yapmak için özel olarak tasarlanan ve temel bir veri işleme birimi veya işlem parçası (İP) (processing element (PE)) şeklinde adlandırılabilecek bir öbeğin çoklanması ile tasarlanan yapılardır. Sistolik diziler yapıları gereği çok yüksek işlem kapasitesine sahiptir. Ancak, bu işlemler tek düzedir ve çok fazla çeşitlilik göstermezler. Literatürde, bir boyutlu, iki boyutlu ve üç boyutlu sistolik dizi kullanan çalışmalar önerilmiştir [10,11,12]. Sistolik dizilerin çok hızlı ve ölçeklenebilir olmaları avantajlı taraflarıdır. Ancak bunun yanında, bu yapılar donanım karmaşasını arttırabilir ve tasarlanmaları zahmetlidir. Ayrıca bu yapılar sadece uygulamaya özgü çalışabilirler [13,14].

1.4 Tezin Kapsamı ve Katkısı

Bu tez çalışmasında video kodlama işleminin hesap yükü en yüksek kısmı olan hareket kestiriminin, donanımsal olarak etkin şekilde gerçeklenmesine yönelik yeni yöntemler ve donanım mimarileri önerilmiştir. Literatürdeki çalışmalarda HK işlemi çoğunlukla 8bit/piksel derinliğindeki video çerçeveleri üzerinde gerçekleştirmektedir. Yakın geçmişte daha düşük bit derinlikleri kullanan yeni hareket kestirimi yaklaşımları önerilmiştir. Ancak bu yaklaşımlar için uygun donanım mimarileri üzerinde yeterli çalışma yapılmamıştır. Bu tez çalışmasında ele alınan temel konu düşük bit gösterimi kullanan HK yaklaşımlarını etkin şekilde gerçekleştirebilecek donanım mimarilerinin geliştirilmesidir. Bunun yanı sıra düşük bit gösterimi kullanan özgün hareket kestirimi yaklaşımları, donanım mimarisi de göz önüne alınarak önerilmiştir.

Tasarlanan mimarilerin doğrulanması için, geliştirilen hareket kestirimi algoritmalarının bilgisayarda koşturulması ile elde edilen hareket vektörleri, benzetim sonucunda donanımın ürettiği hareket vektörleri ile karşılaştırılmıştır. Hareket kestirimi algoritmaları C dili ile kodlanmıştır. Donanımın benzetimi için Mentor Graphichs ModelSim yazılımı kullanılmıştır ve donanım tasarımında Verilog donanım tanımlama dili kullanılmıştır.

Tez kapsamında yapılan çalışmalar temel olarak üç kısımda incelenebilir:

- Literatürde hali hazırda önerilmiş olan iki farklı düşük bit gösterimi temelli hareket kestirimi yaklaşımı öncelikle literatürdeki düşük bit gösterimi kullanan bir donanım mimarisi ile gerçekleştirilmiştir. Bunun yanı sıra literatürde bulunan fakat daha önce düşük bit gösterimi temelli bir HK yöntemine uygulanmamış olan bir mimari bu iki HK yaklaşımının gerçeklenmesi için bu tez kapsamında ilk kez kullanılmıştır.

- Literatürde HK'nin işlem yükünü azaltmak için kullanılan yöntemlerden biri de bit-kesme (bit truncation) tekniğidir. Bu yöntem hâlihazırda piksel değerlerinin, doğrudan ikili (binary) karşılıklarındaki en değerliksiz belli sayıda bitin kesilmesi şekilde yapılmaktadır. Bu tez çalışması kapsamında bit kesme işleminin doğrudan piksel değerlerinin ikili karşılıklarından yapılması yerine Gray kodlanmış piksellerden yapılması önerilmiştir. Böylelikle hareket kestirim başarımının önemli ölçüde artırılabildiği gösterilmiştir. Ek olarak, geliştirilen bu özgün yöntem için bir donanım

mimarisi tasarlanmıştır.

- HK doğruluğunu artırmak için yaygın olarak kullanılan bir yol da kesirli hareket kestirimidir (KHK - fractional pel motion estimation). Geçmişte doğrudan 8 bit/piksel derinliğindeki imgeler üzerinde gerçekleştirilen bu yaklaşım düşük bit gösterimi temelli HK için pek incelenmemiştir. Literatürde bu konuda yapılan tek çalışmada KHK orijinal derinlikte piksellere aradeğerleme yapıldıktan sonra ikili şekle dönüşüm yapılmıştır. Bu tez çalışmasında ise tamamen ikili veriler üzerinde çalışan özgün bir KHK yaklaşımı ve bu yöntemin donanım mimarisi önerilmiştir.

Bu tez çalışması kapsamında geliştirilen mimariler Verilog donanım tanımlama dili kullanılarak saklayıcı aktarımı seviyesinde (SAS) (RTL-Register transfer level) kodlanmış ve Mentor Graphics ModelSim benzetim aracı kullanılarak çalışabilirlikleri gösterilmiştir. Xilinx ChipScope Pro aracı kullanılarak mimariler FPGA üzerinde gerçek zamanlı olarak doğrulanmıştır. Önerilen mimarileri tanımlayan SAS kodlar Synplicity Synplify Premier, Xilinx XST gibi farklı sentez araçları kullanılarak sentezlenmiş ve elde edilen sonuçlar karşılaştırılmıştır. Mimariler arasındaki farklar, sahip oldukları donanım karmaşıklığı, güç tüketimleri ve çalışma performansları açısından incelenmiş ve yakın geçmişte önerilen güncel çalışmalarla karşılaştırılmışlardır.

İleriki bölümlerde ayrıntıları verilen deneysel sonuçlarla tez kapsamında önerilen özgün HK yöntem ve mimarilerinin yüksek başarımı gösterilmiştir.

1.5 Tez Planı

Tezin ikinci bölümünde, geçmişte önerilen HK yöntemlerinin geniş bir özeti yapılmıştır. Bu yöntemler, kullanılan HK yaklaşımlarına göre sınıflandırılmıştır. Üçüncü bölümde, geçmişte önerilen HK donanımı mimarileri, sınıflandırılmış ve geniş bir özet yapılmıştır. Dördüncü bölümde, özgün donanım mimarileri tasarlanan düşük bit gösterimi temelli HK yöntemleri anlatılmış ve ardından düşük bit gösterimi temelli iki yöntem için önerilen mimarilerin tasarım adımları sunulmuştur. Mimarilerin çalışması anlatıldıktan sonra geçmişte önerilen çalışmalarla, güç tüketimi, alan ve başarımı karşılaştırılmıştır. Beşinci bölümde bu tez çalışması kapsamında geliştirilen Gray kodlama temelli HK yöntemi tanıtılmış ve tasarlanan donanım mimarisi anlatılmıştır. Altıncı bölümde, güncel video kodlama uygulamalarında mutlaka yer

alması gereken KHK yönteminin, düşük bit gösterimi temelli yöntemler ile nasıl gerçeklenebileceği konusu açıklanmış ve yeni bir yöntem ve donanım mimarisi verilmiştir. Yedinci ve son bölümde, bu çalışma kapsamında önerilen yöntem ve mimariler genel olarak özetlenmiş ve geleceğe dönük ne gibi iyileştirmeler yapılabilir sorusuna yanıt bulmaya çalışılmıştır.

2. HAREKET KESTİRİMİ YÖNTEMLERİ

2.1 Giriş

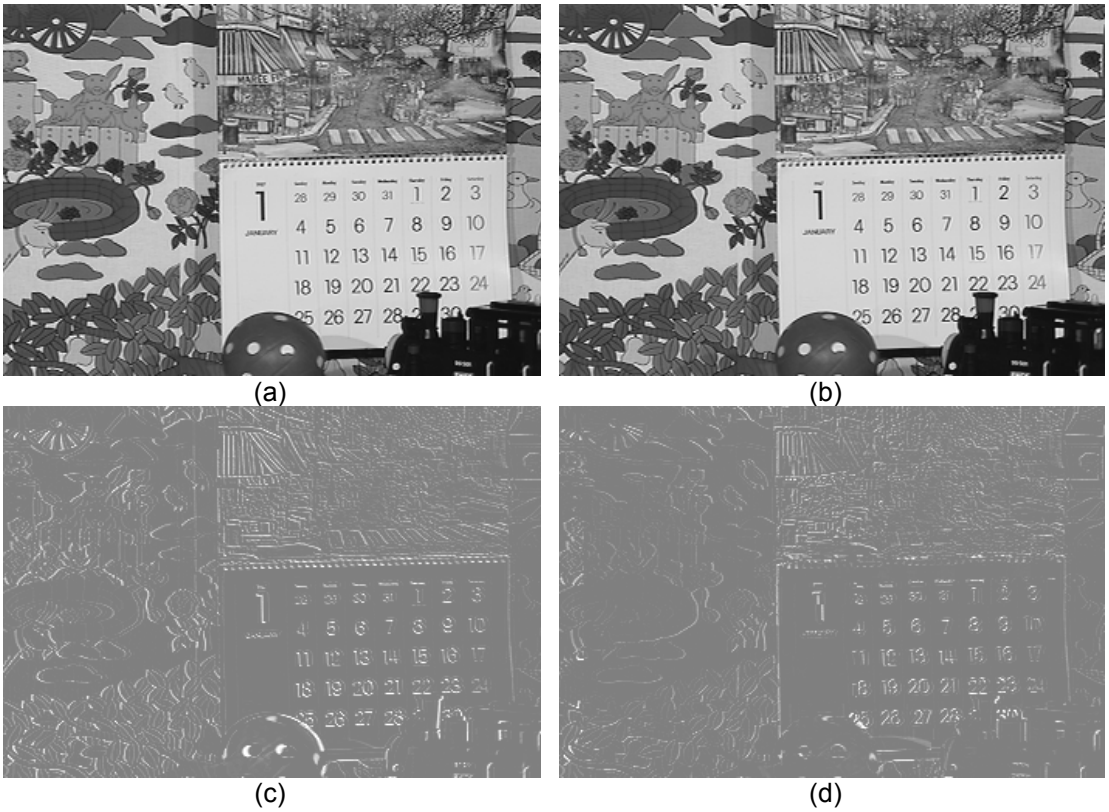
HK işlemi zamansal artıklığı ortadan kaldırmakta olup bunun sonucunda yüksek oranlarda sıkıştırma elde edilebilmektedir. HK işleminin, video kodlayıcı içerisindeki toplam işlem yükünün en büyük bölümünü oluşturduğu (kodlamada kullanılan algoritmaya bağlı olarak %50-90) düşünülürse bu işlemin çok etkin bir şekilde gerçekleştirilmesinin bir zorunluluk olduğu anlaşılmaktadır [14].

Video çerçeveleri her ne kadar RGB gibi üç ana renk uzayında ifade edilebilse de bu gösterim kodlama verimliliği açısından uygun değildir. Video kodlama için yaygın olarak imge çerçeveleri öncelikle RGB uzayından (2.1)'de verilen şekilde YC_RC_B uzayına dönüştürülmektedir.

$$\begin{aligned} Y &= 0.299 \times R - 0.587 \times G + 0.114 \times B \\ C_B &= 0.564 \times (B - Y) \\ C_R &= 0.713 \times (R - Y) \end{aligned} \tag{2.1}$$

HK işlemi tipik olarak işlem yükünü düşük tutmak amacıyla sadece Y çerçeveleri üzerinden gerçekleştirilir. Şekil 2.1'de görülen ilk iki video çerçevesi, *"mobile"* video dizisinden seçilmiş ardışık iki video çerçevesinin Y bileşenidir. Şekil 2.1(c)'de görülen video çerçevesi, Şekil 2.1(a) ve Şekil 2.1(b)'de görülen video çerçevelerinin farkına 128 öteleme (offset) değeri eklenerek elde edilmiştir. Şekil 2.1(d)'de görülen video çerçevesi ise Şekil 2.1(b)'deki video çerçevesinin, HK ile (a)'daki video çerçevesinden imgeden kestirilmesi sonrasında elde edilen kestirim imgesinin Şekil 2.1(a)'daki video çerçevesi ile farkının alınmasıyla elde edilmiştir. Şekil 2.1(d)'deki video çerçevesi için de öteleme olarak 128 kullanılmıştır. Bu imgelerden görülmektedir ki önce HK ve ardından hareket dengelemesi işlemlerini yapmak, kodlanması gereken veri miktarını azaltmaktadır.

Hareket karşılama işleminin yapılabilmesi için öncelikle hareketli bölgelerin hareket miktarlarının kestirilmesi gerekmektedir. Şekil 2.2'de bu işlem basitçe görselleştirilmiştir.

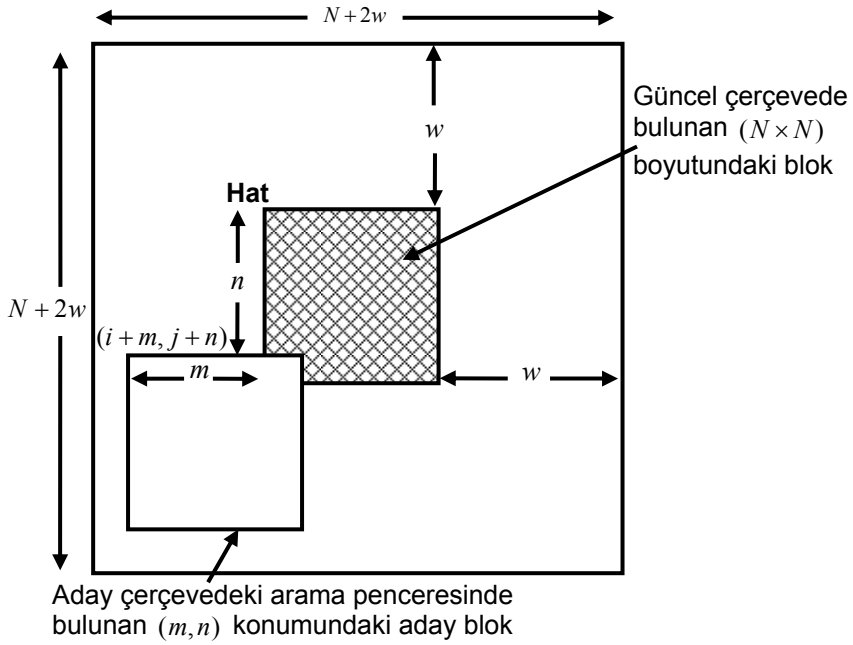


(a) (b)

(c) (d)

Şekil 2.1: a)*"mobile"* dizisi 14. çerçevesi, b) *"mobile"* dizisi 15. çerçevesi. c) Sadece öngörücü kodlama kullanılması durumunda ile kodlanması gereken fark bilgisi (Görselliği iyileştirmek amacıyla 128 değer ötelenmiş hali.), d) Hareket dengelemeli öngörücü kodlama ile kodlanması gereken fark bilgisi(+128)

HK işlemi için önerilmiş farklı yöntemler bulunsa da video kodlama için kabul görmüş yöntem, blok uyumlama (BU) temelli HK'dir. BU temelli HK işlemi genel haliyle belirli büyüklükteki bloklara ayrılan o andaki (kodlanacak) video çerçevesindeki her bir bloğun referans video çerçevesinde büyüklüğü önceden belirlenen bir arama penceresi içerisinde aranması ve her bir aday bölge için elde edilen uyuma bağlı olarak hareket vektörünün belirlenmesinden ibarettir. Tipik bir HK işleminde güncel çerçeve $(N \times N)$ boyutlu, örtüşmeyen (non-overlapping) bloklara bölünür. Daha sonra her bir blok referans (tipik olarak) önceki çerçevede $(N+2p)$ boyutundaki arama penceresinde $[-p,+p-1]$ aralığında toplam $(2p)^2$ farklı konumda aranır. Belirlenen uyum ölçütü doğrultusunda arama işlemi sonunda hareket vektörü hesaplanır. En büyük işaret gürültü oranı (PSNR-Peak signal to noise ratio) değerine göre en iyi başarımı karesel farkların toplamı ölçütü vermektedir. Ancak kare alma işleminden kaçınmak için bunun yerine yaygın olarak (2.2)'de görülen mutlak farklar toplamı (MFT) ölçütü kullanılmaktadır.



Şekil 2.2: HK işleminin $(N \times N)$ blok boyutu ve $\pm w$ arama aralığı için görsel ifadesi

$$MFT(m,n) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} \left| c(i,j) - s(i+m, j+n) \right| \tag{2.2}$$

$$HV = \left\{ (u,v) \middle| MFT(u,v) \le MFT(m,n); -p \le m,n \le p-1 \right\} \tag{2.3}$$

(2.2) eşitliğindeki $MFT(m,n)$ ifadesi, (m,n) konumundaki aday blok ile güncel blok arasındaki mutlak farkı temsil etmektedir. $c(i,j)$, güncel blok piksellerini, $s(i+m, j+n)$ ise (m,n) konumundaki aday blok piksellerini temsil etmektedir. Her iki blok, $N \times N$ boyutundadır. (2.3)'de görülen eşitlikteki HV, $(2p)^2$ kadar aday bölge içerisinde güncel blok ile arasındaki fark en küçük olan aday bloğun göreceli konumunu, yani hareket vektörünü (HV) temsil etmektedir. Belirlenen arama çerçevesindeki bütün aday konumların arandığı HK yaklaşımı tam arama temelli blok uyumlama yöntemi (TABUY) olarak adlandırılmaktadır.

TABUY, olası en iyi uyumlama sonucunu vermekle birlikte hesap yükü oldukça fazladır. Literatürde TABUY'nin başarımını koruyarak işlem yükünü azaltmaya çalışan bir çok yaklaşım önerilmiştir. Hızlı HK yöntemleri olarak bilinen bu yöntemler, genel olarak başarımdan küçük bir miktar ödün vererek hesap yükünü önemli miktarda azaltmaktadırlar. Ayrıca tamsayı HK başarımın artırmak için daha yüksek

başarımlı HK yöntemleri de literatürde önerilmiştir.

2.2 Hızlı Hareket Kestirimi Yöntemleri [14]

Bu bölümde, literatürde mevcut hızlı hareket kestirimi yöntemleri altı sınıfta incelenmiştir. Bunlardan ilk beşi kayıplı, sonuncusu ise kayıpsız yöntemlerdir. Kayıplı yöntemlerde tipik olarak HK başarımı TABUY'ye göre düşüktür. Kayıpsız yöntemlerde ise TABUY ile aynı başarım TABUY'ye göre daha düşük bir hesap yükü ile elde edilmektedir.

2.2.1 Arama noktası seyreltme

TABUY ile elde edilen HK sonuçları en iyi (optimum) olarak değerlendirilirse, bu en iyi HV konumundan uzaklaşıldıkça uyumlama hatasının tekdüze (monotonic) bir şekilde artacağı genel olarak yapılan bir varsayımdır. Bu varsayımdan hareketle bütün olası aday noktaların yerine bu aday noktalardan bazılarının aranması yolu ile arama noktaları seyreltilerek TABUY'nin işlem yükü, başarımdan ödün vermeden azaltılabilmektedir. Bu tip yaklaşımlar seyrek arama noktası temelli yöntemler olarak isimlendirilmektedir. Literatürde önerilen hızlı HK yaklaşımlarının birçoğu arama noktası seyreltme yaklaşımını kullanmaktadır. Logaritmik arama [15], üç adım arama (3AA) (3SS-three step search) [16], türemiş doğrultuda arama (conjugate direction search) [17], geliştirilmiş logaritmik arama (modified logarithmic search) [18], çapraz arama (cross search) [19] bunlardan bazılarıdır. Paralel sıradüzensel tek boyutlu arama (paralel hierarchical one dimensional search) [20], bir boyutlu tam arama (one dimensional full search) [21], yeni üç adım arama (new three step search) [22], dört adım arama (four step search) [23] geçmişte arama noktasını azaltmak amacıyla önerilmiş çalışmalardandır. Merkez meyilli elmas arama (center biased diamond search) [24,25], geliştirilmiş elmas bölgesel arama (advanced diamond zonal search) [26], merkez meyilli uyarlanabilir arama (center biased adaptive search) [27], çapraz elmas arama (cross diamond search) [28] ve öngörülü çizgi arama (predictive line search) [29] bir boyutlu kademeli azalan arama (one-dimensional gradient descent search) [30] yöntemleri de yine arama noktası sayısını azaltan yöntemlerin başarımlarını artırmaya yönelik önerilmiş diğer çalışmalardır. Yakın geçmişte ise altıgen temelli arama yöntemleri ve içeriden arama yaklaşımı adı yeni teknikler yaygın olarak kullanılmaya başlanmıştır. [31]'de klasik elmas arama yaklaşımı yerine altıgen biçimli bir arama örüntüsü (search pattern) kullanan yeni bir

yöntem ile elmas aramadan yüksek bir başarımın daha az arama noktası ile sağlanabildiği gösterilmiştir. [32]'de bloklar arası ilişkiyi de dikkate alan, altıgen tabanlı bir arama yöntemi geliştirilmiştir. Bu yöntem ayrıca akıllı bir içeriden arama tekniği kullanarak arama noktası sayısını da azaltmaktadır. [33]'de içeriden arama yaklaşımı ve komşu blokların hareketini dikkate alan yeni bir elmas arama yaklaşımı ile klasik elmas arama yaklaşımının başarımının artırılabildiği gösterilmiştir. [34]'deki çalışmada [31, 32]'de kullanılan altıgen arama yaklaşımı daha da geliştirilerek H.264/AVC'de etkin şekilde çalışması sağlanmıştır. Etkin çalışmayı sağlamak için birçok video dizisi test edilip hareket vektörlerinin genel dağılımına göre uygun yeni altıgen yapılar kullanılmıştır. [35]'deki çalışmada ufak hareket vektörlerinin çapraz şekilli bir arama örüntü ile hızlı şekilde bulunabilmesi için bir ön arama, sonrasında ise gerekirse tipik altıgen arama yaklaşımının kullanılması önerilmiştir. Böylelikle arama noktası sayısının azaltılması sağlanmıştır. [36]'da ise daha gelişmiş bir içeriden arama yaklaşımı kullanılarak [32]'deki altıgen arama yaklaşımının başarımı artırılmıştır. [36]'da ayrıca önerilen içeriden arama yaklaşımı elmas aramaya da uygulanarak yöntemin elmas aramanın da başarımını artırdığı gösterilmiştir. [37]'de değişken adım uzunluklu arama kullanan yöntem, uyarlanır bir arama aralığı kararı vererek tam aramaya göre başarımdan fazla bir ödün vermeden hızlı bir arama gerçekleştirmektedir. [38]'de ise hareket vektörü bulunurken sadece bir arama örüntü yerine elmas arama ve logaritmik aramanın uygun yerlerde kullanılması ile başarımda artış sağlanabileceği gösterilmiştir.

2.2.2 Uyum ölçütünün sadeleştirilmesi

MFT uyumlama ölçütü, aday blok ve şimdiki blok içerisindeki bütün pikselleri içerir. [39]'da hesap yükünü azaltmak amacıyla bir alt örnekleme yapısı önerilmiştir. Yatayda ve düşeyde her iki pikselden birisi hesaba katılarak toplam hesap yükü dörtte bir oranında azaltılmıştır. Girişim etkileri ise alçak geçiren süzgeç ile ortadan kaldırılmıştır. [27,41]'de farklı arama konumları için farklı alt örnekleme yöntemleri uygulanmış ve bu sayede girişim etkileri alçak geçiren süzgeç olmadan giderilmeye çalışılmıştır. [42]'de uyarlanabilir piksel-kırımı yöntemi önerilmiştir. Bu yöntemde herhangi bir ilk bölümlendirme gerekmemekte ve her piksel, uyumu belirlemek için gerekli özellikleri içeriyorsa kullanmaktadır. Piksel farkı sınıflandırma (PFS) yöntemi [43] pikseller arasındaki mutlak farkları değerlendirerek uyumlu veya uyumsuz kararının verilmesini sağlamaktadır.

En iyi aday bölge, en fazla sayıda uyumlu pikseli içerir. Bu yaklaşımla, donanım karmaşası büyük oranda azalmaktadır, çünkü toplayıcı yapıları basit sayıcılar ile yer değiştirmektedir. Diğer taraftan, eşik değeri değiştikçe kalite çok değişir ve değerinin ne olacağına otomatik olarak karar vermek kolay değildir.

En küçük-en büyük ölçütü [44] öncelikle bir aday bölgedeki tüm pikseller içinde en büyük mutlak farkı hesaplar. Aynı işlemi bütün aday bölgeler için gerçekleştirdikten sonra mutlak farklar arasında en küçüğe sahip aday bölgeyi HV konumu olarak belirler. Bu yöntemde toplam hesap yükünde bir azalma olmasa da 16 bit toplayıcı yerine sekiz bit karşılaştırıcı kullanmak donanım karmaşasını yaklaşık olarak %15 oranında azaltmaktadır.

[45,46]'de tümlev izdüşümü (integral projection) yöntemi önerilmiştir. Uyumlama için piksel bilgisi yerine her bir satırın ve sütunun toplamına dayanan bir izdüşüm hesaplanır ve ardından bu izdüşümlerin uyumlanması ile HK işlemi yapılır. Bir aday konumdaki tümlev izdüşümü sayısı piksel sayısından daha az olduğu için uyumlama işlemi için gereken hesap yükü oldukça azalır.

2.2.3 Bit-derinliği azaltma

Video çerçevelerinde her bir pikselin ışıklılık değeri tipik olarak 8-bit ile gösterilmektedir. [47]' de her pikselin ilgili blok ortalaması kullanılarak video çerçeveleri ikili hale getirilmektedir. Ardından elde edilen ikili video çerçeveleri kullanılarak yüksek uyum gösteren aday hareket vektörü konumları belirlenmektedir. Sonrasında bu konumlar için MFT hesaplanmaktadır. 1-Bit dönüşümü (1BD) [48], sıradüzensel 1BD [49], 2-bit dönüşümü (2BD) [50], çarpmasız 1BD (Ç1BD) [51] ve kısıtlanmış 1BD (K-1BD) [52] yöntemi gibi donanımsal gerçeklemeye uygun yöntemler yakın geçmişte önerilmişlerdir. [48]'de, çoklu bant geçiren bir süzgeç yapısı kullanılarak ikili hale dönüştürülen video çerçeveleri üzerinde yapılan HK [47]'de önerilen HK yönteminin başarımı artırılmış ayrıca bir de donanım mimarisi sunulmuştur. [48]'de, güncel blok pikseli ile aday blok pikseli arasındaki benzerliğin ölçülmesi için hesap yükü fazla olan mutlak fark alma işlemi yerine donanımsal olarak daha basit bir şekilde gerçeklenebilen mantıksal ÖZEL-VEYA işlemi kullanılmaktadır. Bu yaklaşımla HK başarımından bir miktar ödün verilerek, daha az

donanım, daha az güç tüketimi ve daha yüksek çalışma frekansı elde edilebileceği aşikârdır. [53]'de K-1BD yöntemi öngörücülü altıgen arama yöntemiyle birleştirilmiş ve hesapsal yük bu şekilde daha da azaltılmaya çalışılmıştır. [54]'de bit kesme işlemi ile piksellerin ikili gösterimlerindeki belli sayıdaki en değerliksiz bit atılarak piksellerin bit-derinliği azaltılmıştır. [54]'de, 4 bite kadar kesme yapılarak gerçekleştirilen HK işlemi doğruluğunda önemli bir kayıp olmadığı ifade edilmektedir. Piksel kesme işlemi hesap yükünün düşürülmesine ve dolayısıyla güç tüketiminin azaltılmasına olanak sağlamaktadır.

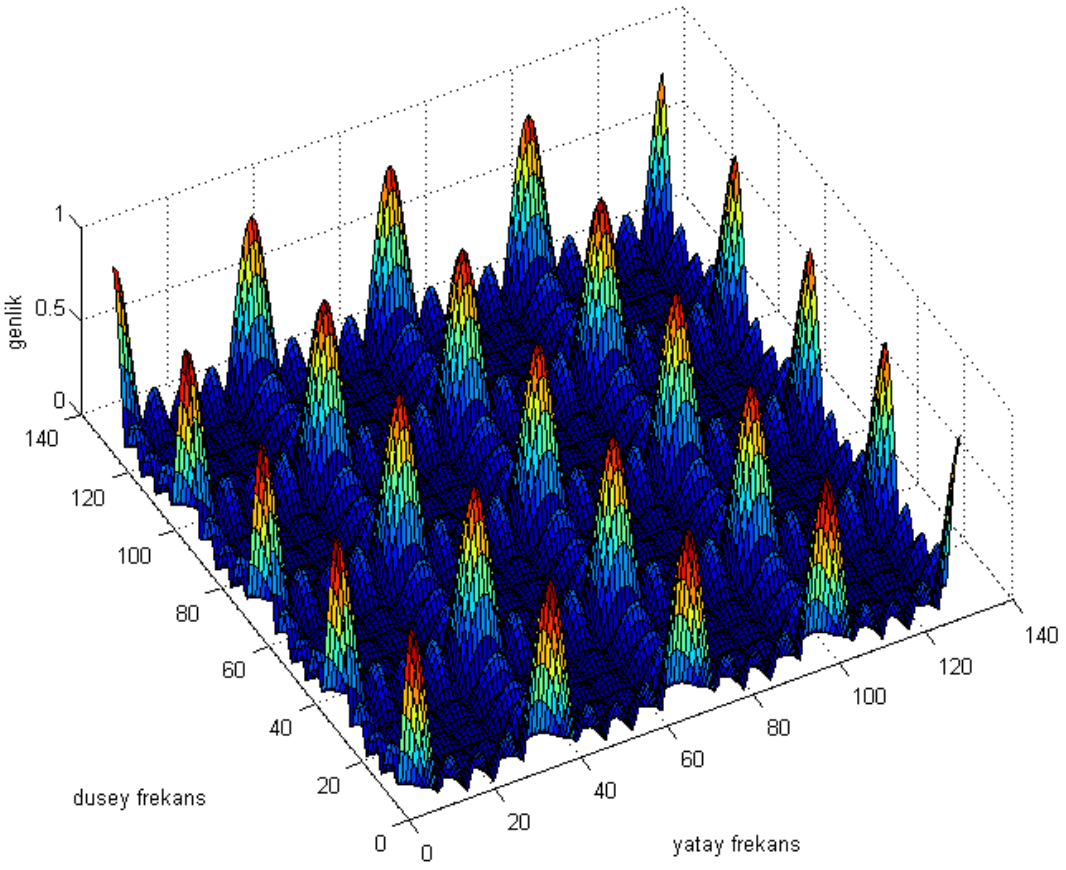
2.2.3.1 Bit-düzlemi uyumlama yöntemi

Bit-düzlemi uyumlama (BDU) yöntemi ilk olarak [47]'de tam arama işlemine bir ön işlem olarak önerilmiştir. Bu çalışmada bit düzlemleri elde edilirken blok ortalamaları eşik olarak kullanılmıştır. [47]'de önerilen 1BD işlemi (2.4)'de görüldüğü şekilde gerçekleştirilir.

$$B(i,j) = \begin{cases} 1, & eğer\ I(i,j) \geq t_{bm} \\ 0, & diğer \end{cases} \tag{2.4}$$

Burada t_{bm} eşik olarak kullanılan blok ortalamasını, (i,j) piksel indislerini ve $I(i,j)$' de 8bit/piksel ile gösterilen video çerçevesinde ilgili indiste bulunan pikseli temsil etmektedir. [48]'de, 1BD işlemi için blok ortalaması yerine asıl video çerçevesinin çoklu bant geçiren bir süzgeç ile süzgeçlenmesi ile elde edilen yeni imgedeki karşılık gelen pikseller eşik olarak kullanılmıştır. [48]'de kullanılan 17×17 boyutlu çekirdekten oluşan çoklu bant geçiren süzgecin çekirdek yapısı (2.5)'de görülmektedir. Bu çekirdeğin frekans yanıtı Şekil 2.3'de görülmektedir.

$$K(i,j) = \begin{cases} 1/25, & eğer\ i,j \in [0,4,8,12,16] \\ 0, & diğer \end{cases} \tag{2.5}$$



Şekil 2.3: [48]'de önerilen çoklu bant geçiren süzgecin frekans yanıtı

Bu durumda 1BD işlemi (2.6)'da görüldüğü gibi gerçekleştirilmektedir.

$$B(i,j) = \begin{cases} 1, & eğer\ I(i,j) \geq I_F(i,j) \\ 0, & diğer \end{cases} \tag{2.6}$$

Burada $I_F(i,j)$, $I(i,j)$ imgesinin K konvolüsyon çekirdeği uygulanarak süzgeçlenmiş halini temsil etmektedir.

[48]'de video çerçeveleri arasındaki uyumun hesaplanmasında uyumsuz nokta sayısı (UNS) (NNMP – Number of Non-Matching Points) ölçütü kullanılmaktadır. Bu ölçüt (2.7)'de verilen şekilde hesaplanmaktadır. En küçük UNS değerini veren aday blok konumu hareket vektörü olarak belirlenir.

$$UNS(m,n) = \sum_{i=0}^{M-1} \sum_{j=0}^{M-1} B^t(i,j) \oplus B^{t-1}(i+m, j+n) \tag{2.7}$$
$$-s \leq m,n \leq s-1$$

2.2.3.1.1 Eşikleme

(2.5)'deki çekirdek kullanıldığında her bir pikselin bir bitlik ikili sayıya dönüştürülmesi 24 toplama, bir çarpma ve bir karşılaştırma yani toplamda bir piksel için 26 işlem gerektirir. 720×480 boyutlu bir imge için toplamda gerekli işlem performansı $269.568 Mişlem / saniye$'dir. Burada her işlem süreç olarak bir biri ile özdeş kabul edilmektedir oysa pratikteki durum farklıdır.

2.2.3.1.2 Piksel uyumlama

Blok boyutu 16×16 kabul edilirse her bir aday blok için 256 adet XOR ve 255 adet toplama işlemi gerekmektedir. $32 bit$ mimari kullanıldığı varsayılırsa her bir XOR komutunda 32 adet bit seviyesinde XOR yapılmaktadır, dolayısıyla 8 komut ile 256 bitlik XOR yapılabilmektedir. Eğer saklayıcılardaki birlerin sayısını sayabilen bir komut olduğu varsayılırsa toplamda 8 komut bu işlem için yeterli olacaktır. Bunun dışında tüm birlerin sayısını hesaplayabilmek için de yedi tane toplama komutuna gerek vardır. Dolayısıyla bir aday bölge için toplamda 23 komut gereklidir.

Eğer saklayıcılardaki birlerin sayısını sayabilecek bir komut yoksa bu işlem okuma tablosu (LUT) ve basit toplama işlemleri ile değiştirilebilir. 256 girişi olan LUT olduğunu ve her bir girişinde, 8 bitlik bir sayıdaki birlerin sayısını tuttuğunu varsayalım. Bu durumda 32 bitlik bir saklayıcıdaki birlerin sayısını 4 adet okuma tablosu ve 3 adet toplayıcı ile hesaplamak mümkün olur. Bu durumda gereken toplam komut sayısı $(8+8\times(4+3)+7)=71$ olacaktır. Kıyaslama olabilmesi açısından ortalama mutlak hata hesabında bir aday bölge için yapılması gereken işlem sayısı $8 \times 256 = 768$'dir. Ortalama mutlak değer hesabında bir çıkarma, bir mutlak değer ve bir de toplama işlemi yapıldığından, toplamda piksel sayısının üç katı işlem gerekir.

Tablo 2.1'de mutlak farklar toplamı (MFT) ve bit uyumlama ölçütünün tam arama yöntemi için gerektirdiği işlem sayısı görülmektedir. Çerçeve boyutu 720×480 piksel, blok boyutu 16×16 piksel ve çerçeve oranı $30 çerçeve / saniye$, arama aralığı da $[-15,15]$ alınmıştır.

BDU32 yukarıda bahsedilen 32 bitlik sistemi temsil etmektedir. BDU için belirtilen işlem sayısı eşikleme yükünü de içermektedir. Tablodan da anlaşılacağı üzere logaritmik aramada BDU ve BDU32 arasında çok az fark vardır çünkü işlem sayısının büyük kısmı ($269.568 Mo / s$) eşikleme için harcanmaktadır [55].

Tablo 2.1: MFT ve BDU ölçütlerini kullanan tam arama logaritmik arama yöntemleri için gereken hesaplama sayısı [55]

Arama Yöntemi	MFT	BDU	BDU32
Tam arama	$29.89 Gişlem / saniye$	$3.03 Gişlem / saniye$	$1.16 Gişlem / saniye$
Logaritmik	$1.02 Gişlem / saniye$	$364.45 Mişlem / saniye$	$300.3 Mişlem / saniye$

2.2.4 Öngörülü arama

2.2.1 bölümünde gösterilen yöntemlerden sezgisel hızlı arama yöntemleri, hızlı hareketli nesneler içeren video dizilerinde düşük HK başarımı gösterirler çünkü tekdüze artan hata modeli sık sık yanlış sonuç üretir. Arama noktalarının azaltılmasını öneren yöntemler ise sık sık yerel minimumlara takılırlar. Öngörülü arama [54,56-60], uzamsal ve zamansal olarak komşu blokların hareket bilgisini kullanarak güncel bloğun hareket vektörünü kestirmeye çalışır. Hesapsal yük ve arama alanını etkin bir biçimde azaltır. [28]'de yapılan çalışmada, sol üst, üst ve sağ üst komşu blokların HV'lerinin, bu HV'lerin ortalamalarının ve sıfır HV yani bir önceki çerçevedeki aynı bloğun HV'sinin bloğun HV'si olabileceği öngörülmektedir. Bu öngörücüler öngörülü aramada en çok kullanılanlardandırlar.

2.2.5 Sıradüzensel Arama

Çoklu çözünürlüklü, piramide benzeyen bir yapıdadır ve görüntü işleme için hesapsal olarak çok güçlü bir düzenlemedir. TABU yönteminin hesabından tasarruf sağlamak için piramit yapıya uygun yeniden sıralama çok yaygındır. Çoklu çözünürlük yöntemi, ilk başta kaba bir HK işlemi ve ardından daha hassas bir seviyede bu kestirimin düzeltilmesidir. Genelde iki veya üç sıradüzensel seviye kullanılmaktadır [61-62]. Bu yöntemde kabaca yapılan HK işlemi sonucunda yerel minimumlara düşme ihtimali vardır, çünkü imge alt örneklenmektedir. Aslında bu yöntem blok uyumlama yönteminde en etkin yöntemlerden birisi olarak nitelendirilmektedir ve büyük çerçeve boyutlarında ve geniş arama aralıklarında en sık kullanılan yöntemdir.

2.2.6 Erken Sonlandırmalı Tam arama

Bu tip yöntemlerde basit bir kontrol ile bir aday bölgenin en uygun olup olmadığına karar vermeye çalışılır, ardından daha ayrıntılı bozulma (distortion) hesabı için sadece olası aday bölgelerde arama yapılır. Sonuç olarak imkânsız aday bölgeler için yapılması gereken çok fazla miktardaki gereksiz hesaplama işleminden kurtulmak mümkün olmaktadır. Ardışık eleme yönteminde (AEY) (successive elimination algorithm) [64] aday blok ile güncel bloğun piksellerinin toplamının mutlak farkı o ana kadar hesaplanan MFT'nin en küçük değerinden daha küçükse o zaman bu aday blok için hesap yapılmaz. AEY'de, yapılmasına gerek olmayan işlemlerin atlama oranının artırılması için ilk HV değerinin iyi tahmin edilmesi gerekmektedir. Bu nedenle AEY, genelde hareket öngörücülerle veya sarmal (spiral) arama yöntemiyle birlikte kullanılır. Çok seviyeli AEY (ÇSAEY – Multi Level SEA) [65-67] klasik AEY yöntemine göre kontrol yordamını değiştirerek aday bölge atlama oranını büyük miktarda artırabilmektedir. [68]'de ÇSAEY yöntemi tek komut-çoklu veri yöntemi ile birleştirilerek daha da geliştirilmiştir ve bu sayede yöntemin hızı daha da artırılmıştır.

Kısmi bozulma elemesi yöntemi (KBEY) (partial distortion elimination algorithm) basit ama etkili bir yaklaşımdır[69]. Bir aday bölgenin kısmi bozulma değeri o ana kadar hesaplanmış olan en küçük bozulma değerinden büyükse bu aday bölge için hesap yapmaya gerek yoktur. Klasik KBEY'de tam arama ile aynı HK başarımı elde edilir. HK başarımından bir miktar taviz vererek hesap yükünü azaltan KBEY yaklaşımları da mevcuttur. [70]'de kısmi bozulma ve minimum bozulma değerleri normalize edilerek arama işleminin erkenden sonlandırılma sayısı artırılmaktadır. Buna ek olarak kısmi bozulmalar gruplandırılarak ve sarmal arama yöntemi kullanılarak erken sonlandırma oranı daha da artırılmaktadır. Küçük miktarda doğruluk kaybına sebep olmasına rağmen çok yüksek çalışma performansına sahiptir. [71]'de tarama sırasının uyarlanabilir yapılması ile KBEY'nin başarımının daha da artırılabileceği önerilmiştir. [73]'de, verilen bir kısmi bozulma için toplam bozulmanın alacağı değerlerin olasılık dağılım fonksiyonu sunulmuştur. [74]'de KBEY'yi geliştirmek için erken sonlandırma işleminin zamanlamasının nasıl olması gerektiği sorusuna analitik bir çözüm önerilmiştir.

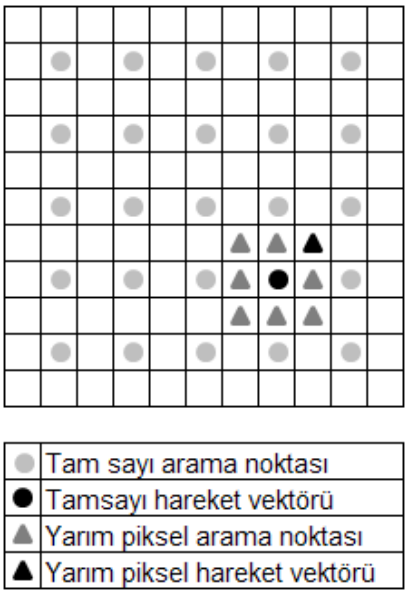
Kazananı güncelleme yöntemi (winner update) [74], temel olarak, en küçük kısmi MFT'yi seçmeyi amaçlar. Bu yöntemde ilk olarak her aday bölgenin ilk pikseli için

uyumlama hatası hesaplanır. Sonrasında en küçük hatayı veren aday için ikinci pikselin uyumlama hatası hesaplanıp önceki uyumlama hatasına eklenir. Bu aşamada elde edilen toplam hata en küçük olduğu sürece bu işlem devam eder. Aksi takdirde o anda en küçük toplam hataya sahip olan aday konum için sıradaki pikselin hatası hesaplanır. Bu şekilde devam ederek bütün pikselleri için uyumlama hatası hesaplanan aday konum hareket vektörü olarak belirlenir.

2.3 Yüksek Başarımlı Hareket Kestirimi Yaklaşımları

2.3.1 Kesirli HK

Blok uyumlama yönteminde arama penceresinde tam sayı adımlarla ilerlemek bir zorunluluk değildir. Bu noktada kesirli HK (KHK) kavramı ortaya çıkmaktadır. HK işleminin doğruluğunu bir adım daha ileri götürmenin bilinen yollarından bir tanesi, tamsayı arama aralığından kesirli arama aralığına inmektir. Kesirli HK (KHK) yapılabilmesi için kesirli çözünürlükteki piksellerin elde edilmesi gerekmektedir. Bu piksellerin elde edilmesi için imge çerçevesindeki mevcut pikseller üzerinde aradeğerleme işlemi yapılması en yaygın tercih edilen yöntemlerdendir. Bunun için, sıklıkla çifte doğrusal aradeğerleme (bilinear interpolation) yöntemi kullanılır [8]. $1/K$ adım aralığında bir KHK yapılmak istendiğinde, imge çerçevesinde K katlık bir aradeğerleme işlemi uygulanmalıdır. Şekil 2.4'de $K=2$ için örneklenmiş bir KHK sahnesi görülmektedir. Bu durum, yarım piksel arama olarak bilinmektedir.



Şekil 2.4: Yarım piksel arama işlemi

Şekilden 2.4'den de görüldüğü üzere, KHK işleminde ilk önce tamsayı arama işlemi gerçekleştirilir ardından tamsayı hareket vektörünün etrafındaki kesirli noktalarda yeniden arama işlemi gerçekleştirilir.

2.3.3 Değişken Blok Boyutlu Arama

Değişken blok boyutlu HK (DBBHK) yeni bir kodlama yöntemidir ve başarımı artıran, geleneksel sabit blok boyutlu HK (SBBHK) yöntemlerine kıyasla daha etkin kestirim yapılmasını sağlar. SBBHK işleminde eğer bir MB içerisinde farklı hareket yönünde iki nesne varsa HK başarımı düşer. Diğer taraftan DBBHK'de aynı şartlarda MB daha küçük bloklara bölünerek bu durum giderilebilir. Sonuçta kodlama başarımı artar. DBBHK yöntemi güncel video kodlama standartlarından H.263 [6], MPEG-4 [4], WMV9.0 [75], ve H.264/AVC [7]'de kullanılmaktadır. Örneğin H.264/AVC'de bir MB 4×4, 4×8, 8×4, 8×8, 8×16, 16×8 ve 16×16 boyutlu alt bloklar şeklinde işlenebilir. DBBHK işlemi daha yüksek oranlarda sıkıştırma gerçekleştirmesinin yanında hesap yükü çok fazladır. Dolayısıyla bu yaklaşım, donanım mimarisi tasarımını da zorlaştırır.

Ayrıca, günümüzde en güncel video kodlama standardı olan H.264/AVC'de, daha iyi bir kestirim sağlamak üzere HK işleminin birden fazla referans çerçevesinde yapılması önerilmektedir [7].

2.3.2 Diğer Yöntemler

Geçmişte, KHK dışında frekans uzayında gerçekleştirilen bir takım işlemler ile HK başarımını artırmaya çalışan yöntemler de önerilmiştir.

Tam sayı HK işlemi için pratik uygulamalarda en yüksek başarım tam arama ve MFT uyumlaması yapılarak elde edilmektedir. Bunun ötesine geçebilmek için arama işleminin kesirli aralıklarda gerçekleştirilmesi en yaygın kullanılan çözümdür. Geçmişte KHK'den başka, HK başarımını artırmak için kodlayıcıların yapılarındaki bazı özellikleri dikkate alarak yapılabilecek iyileştirmeleri değerlendiren çalışmalar da önerilmiştir. Buna en güzel örnek kodlayıcı içerisinde bulunan ve asıl imge ile hareketi karşılanmış imge arasındaki farkı kodlayan dönüşüm kodlayıcısıdır. Geçmişte önerilen pek çok çalışma dönüşüm kodlayıcıyı göz ardı etmektedir. [76]'da bu duruma dikkat çekilmiştir. [76]'ya göre, ortalama mutlak fark (OMF) kullanılarak yapılan HK işleminde artığın kaç bit ile kodlanacağı düşünülmemekte, bu işlem dönüşüm kodlayıcısına bırakılmaktadır. [76]'da HK ile dönüşüm kodlaması arasındaki bu uyumsuzluğu gidermek için özgün bir blok uyumlama yapısı önerilmiştir. Pürüz kısıtlamalı uyumlama metriği (OMF_{PK} smooth constrined MAD), fark bloğu içerisindeki en büyük ve en küçük artıklık farkını dikkate alır. MMD (Max-Min Difference) olarak isimlendirdiği bu fark verisini OMF ölçütüne dâhil eder. Bu durumda oluşan yeni OMF ölçütü (2.8)'de görülmektedir.

$$OMF_{PK} = OMF + \alpha \sum_{m=1}^{4} MMD_m \tag{2.8}$$

Burada görülen α parametresi ağırlıklandırma katsayısıdır. MMD_m de 16×16 boyutundaki makrobloğun (MB) m. 8×8 boyutlu bloğunun MMD ölçütünü temsil etmektedir. Burada işlem yükünün arttığı görülse de, [76]'da bu işlem sonucunda sabit bit oranında video kalitesinin arttığı gözlenmiştir.

[77]'de ise dönüşüm kodlayıcısının yapısına farklı bir noktadan vurgu yapılmıştır. Dönüşüm kodlayıcısında ayrık kosinüs dönüşümü en yaygın kullanılan dönüşüm işlemidir. Burada kullanılan dönüşüm katsayıları ile nicemleme esnasında düşük frekans bileşenlerindeki kayıp düşük olmasına karşın yüksek frekans bileşenlerinde nicemleme aralığı nispeten geniş olduğu için bilgi kaybı artmaktadır.

[77]'de, bu gerçekten yola çıkılarak uyumlama işleminin yüksek frekans bileşenlerinde gerçekleştirilmesi ve düşük frekans bileşenlerinin farkı için ise işin dönüşüm kodlayıcısına bırakılması önerilmiş ve başarılı sonuçlar elde edilmiştir. Önerilen yönteme de kısaca yüksek frekans bileşeni uyumlaması (YFBU) denilmiştir.

[78]'de geleneksel blok uyumlama yöntemi sonucunda elde edilen aday bloğa, uyumlama başarımını artırmak amacıyla frekans uzayında bükme işlemi uygulanmaktadır.

3. HAREKET KESTİRİMİ DONANIMI MİMARİLERİ

3.1 Giriş

HK'nin gerektirdiği çok fazla sayıdaki işlemin (bu sayı H.264/AVC kodlayıcı için yaklaşık 315 $Gkomut/saniye$) gerçek zamanlı yapılabilmesi için farklı alternatifler bulunmaktadır. Bunlardan bir tanesi, sayısal işaret işleme yongalarıyla bu sorunu çözmeye çalışmaktır. Bu yöntem çok esnek bir çözüm gibi görünse de gereksiz güç tüketimi ve nispeten düşük paralel işlem kapasitesine sahiptir. Uygulamaya özel tümleşik devreler (ASIC), güç/hız/alan bakımından en uygun çözüm gibi görünmektedir ancak yeniden düzenlenebilirliğe kapalı oldukları için ilk-örnek geliştirme aşamasında yüksek maliyet getirir. Bu noktada alan programlanabilir kapı dizilerinden (FPGA) faydalanmak ilk örnekleme aşamasında uygun maliyetli ve esnek bir çözüm olarak karşımıza çıkmaktadır. Bazı üst seviye FPGA yongaları, içlerinde adanmış mikroişlemci/mikroişlemciler de barındırabilmektedir.

Hızlı arama yöntemleri, bilgisayar ortamında çalışma zamanını önemli ölçüde kısaltırken silikon üzerinde kapladıkları alan tam arama temelli donanım mimarilerine oranla nispeten küçük olmaktadır [14]. Bu yöntemlerim yapısından kaynaklanan istisnai durumların ise ASIC veya FPGA üzerinde gerçeklenmeleri çok pratik değildir. Bu yöntemler, tekdüze olmayan yapıları nedeniyle kontrol öbeklerinin tasarımını zorlaştırmaktadır. Bu noktada, donanım tasarımı ile ilgilenen araştırmacılar "algoritmalar donanıma nasıl daha uygun hale getirilebilir?" sorusunun cevabını aramaktadırlar. Burada "donanıma uygunluk" kavramı şu şekilde detaylandırılabilir;

- Tasarlanan mimari basit olmalıdır
- Tasarlanan mimari çok yüksek işlem kapasitesine sahip olmalıdır.
- Yonga üzeri (YÜ)(on chip) bellek büyüklüğü mümkün olduğunca az olmalıdır.
- Verinin etkin biçimde işlenmesine olanak sağlamalıdır.

Tasarlanan mimarinin basit olması iki önemli etkene bağlıdır. Bunlardan birincisi tasarlanan donanımın mümkün olduğunca az yer kaplaması, ikincisi de yapılan tasarımın tekdüze ve ölçeklenebilir, yani temel bir takım öbeklerin çoklanması yöntemiyle kolayca gerçeklenebilir olmasıdır.

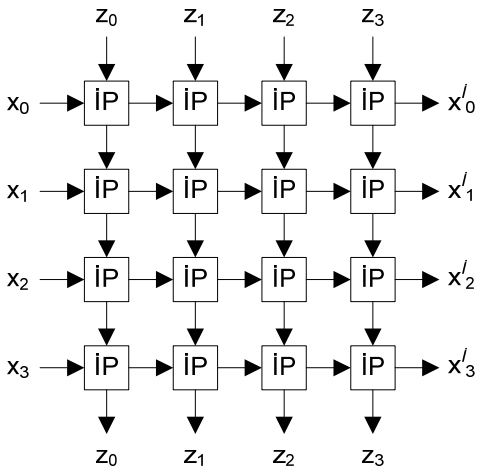
Tasarlanan mimarinin işlem kapasitesinin yüksek olması aynı anda - aynı saat darbesinde- birden fazla işlemi paralel olarak yapılabilmesidir. Bununla birlikte, büyük oranda veri akışına ve işlem karmaşasına sahip olan blok uyumlaması gibi algoritmaların gerçek zamanlı çalışabilmeleri sadece paralel ve ardışık düzende işleme yöntemleri ile mümkün olabilir [79]. Mimarilerin tekdüze yapıları ve birimsel oluşları tasarım maliyetini düşürür.

YÜ bellek büyüklüğünün az olması, donanımın az yer kaplamasının yanında güç tüketiminin de düşük olması anlamına gelmektedir. Zira büyük bir bellekten veri okumak, küçük bir bellekten veri okumaya göre daha fazla güç tüketimine sebep olmaktadır. Günümüzde, belleklerin video kodlayıcı mimarilerinde kullanılması ile ilgili öne çıkan yeni bir yaklaşım da bu öbeklerin YÜ'de tek bir öbek halinde değil de sıra düzensel olarak ayrı ayrı yerleştirilmeleridir. Bu yöntem, sayısal işaret işleme yongalarında ve genel amaçlı işlemci yongalarında hali hazırda uygulanmakta olan ve başarım üzerinde doğrudan etkisi olan bir yöntemdir (örneğin; L1, L2 ön bellek yapıları). Bu sayede nispeten büyük belleklerden/belleklere, az miktarda veri okunmasını/yazılmasını gerektiren uygulamaların fazladan güç tüketiminin önüne geçilebilir. Böylece güç tüketiminde büyük bir azalma elde edilebilir. [9]'da bir video kodlayıcının toplam okuma yazma miktarının 471 $GByte / saniye$ olduğu ve bunun yaklaşık 365 $GByte / saniye$'lik bölümünün tamsayı HK (THK) işlemi için kullanıldığı tespit edilmiştir. Buradan video kodlayıcının toplam bellek erişim oranının önemli bir kısmını HK işleminin yaptığı açıkça görülmektedir.

Bilindiği gibi blok uyumlama tabanlı HK işleminde bir arama penceresi içerisindeki aday bölgelerle uyumluluk ölçütüne bakılarak arama işlemi gerçekleştirilir. Her bir aday bölge arasında çok sayıda ortak piksel vardır. Dolayısıyla bu piksellerin tekrar-tekrar okunması güç tüketimini artırmakta ve performansı düşürmektedir. Geçmişte bu verilerin yeniden kullanılmasını (VYK) (data reuse) sağlayan birçok yöntem önerilmiştir. VYK ile ilgili detaylı bir çalışma [9]'da farklı durumlar için incelenmiştir. Bu çalışmaya göre, THK işleminin, önbelleği olmayan bir RISC (Reduced Instruction Set Computer) işlemci ile yapılması durumunda gereken yonga dışı (off chip) bellek

bant genişliği 138.4 *GByte / saniye* olmaktadır. Aynı rakam, ön bellek içeren bir RISC işlemci için 866.6 *MByte / saniye*'ye düşmüştür. Ancak, bu defa YÜ bellek bant genişliği 138.4 *GByte / saniye*'ye çıkmıştır. Bu, tercih edilen bir durumdur çünkü yonga dışında bir bellekten veri okumaktansa YÜ bir bellekten veri okumak daha hızlı ve güç tüketimi daha az olan bir işlemdir. Özetle, [9]'da önerilen iyileştirmelerle birlikte yukarıda bahsedilen rakamlar YÜ bellek bant genişliğinde %99.9, yonga dışı bellek bant genişliğinde ise %89.6 oranında azaltılabilmiştir. Bu rakamlar yoğun işlem yükü olan uygulamalarda ASIC/FPGA tasarımların ne gibi avantajlar getirebileceği konusunda gayet çarpıcıdır.

Sistolik diziler ilk olarak H.T.Kung tarafından önerilmiştir [79,80,81,82]. Sistolik diziler, özdeş, temel İP'lerin tekdüze bir biçimde ardışık düzende çalışabilecek şekilde bir araya getirilmeleri ile oluşturulmaktadır. Şekil 3.1'de bir sistolik dizi yapısı görülmektedir.



Şekil 3.1: Sistolik dizi yapısı

Sistolik diziler geçmişte pek çok uygulamada kullanılmakla beraber TABUY için yaygın olarak kullanılmaktadırlar. Çünkü bu yapılar kullanıldığında tamamen paralel işlem yapabilme, çok az kontrol sinyali ile denetleyebilme ve tüm bileşenler arasında gerekli verileri yerel bağlantılarla dolaştırarak global bağlantı sayısını azaltmak mümkündür.

Literatürde önerilen donanım mimarileri için iki farklı sınıflandırma yapmak mümkündür. Bunlardan birincisi mimarinin benimsediği arama yöntemine göre diğeri ise mimari içierisindeki işlem parçalarının çalışma şekline göre yapılabilir.

Mimarinin benimsediği arame yöntemine göre yapılacak sınıflandırma kendi içinde ikiye bölünebilir:

1. Tam arama temelli BU yöntemini kullanan donanım mimarileri.

2. Hızlı blok uyumlama mimarileri.

Mimari içerisindeki İP'lerin çalışma şekline göre yapılacak sınıflandırma da kendi içinde ikiye ayrılabilir:

1. Hareket vektörü temelli doğrusal diziler.

2. Kaynak piksel temelli doğrusal diziler.

3.2 Arama Yöntemine Göre Donanım Mimarileri

3.2.1 TABU Donanımı Mimarileri [14]

Birçok TABU donanımı mimarisi geliştirilmiştir. Bu alt bölümde literatürde önerilmiş TABU donanımı mimarilerine kısaca değinilecektir. Önerilen mimarilerin büyük çoğunluğu, yerel olarak bağlanmış İP'lerden oluşan sistolik diziler üzerine inşa edilmiştir. Sıralı düzende ardı ardına akan veriler için herhangi bir denetim birimi tasarımına gerek yoktur. Sürülmesi gereken düşük kapasiteler sayesinde yüksek frekanslarda, dolayısıyla daha yüksek işlem hızlarında çalışma şansı oluşur. Dahası, bellekten okunan veri, İP'ler arasında gezdirilerek bellek bant genişliği büyük oranda düşürülebilir. TABU mimarisinde her bir İP, bir güncel blok pikseli ile bir aday blok pikseli arasındaki mutlak farkı hesaplamakla yükümlüdür. Blok boyutu ve arama aralığı için sırasıyla $N \times N$ ve $[-p, p-1]$ gösterimi kullanılacaktır.

[10]'da Komarek ve Pirsch, sistolik TABU mimarilerinin elde edilmesini sağlayan ayrıntılı bir haritalama yöntemi önermişlerdir. Bu yöntemi kullanarak geliştirilen dört farklı mimari tamamen sistolik olup herhangi bir genel bağlantı içermez ancak, kullanılan bellek veri hattı çok geniştir.

[83]'de Vos ve Stegherr, yarı sistolik dizi ve toplayıcı ağacından oluşan bir mimari önermişlerdir. Bu çalışmanın en önemli katkısı, arama sıralamasındaki değişiklik olan yılan (snake) aramadır. Donanım mimarisinde kullanılan toplam İP sayısı N^2 'dir ve güncel blok pikselleri, karşılık düşen İP'de saklanır. İhtiyaç duyulan geniş saklayıcı kümesi, bu tasarım için bellek bit genişliği ile bir ödünleşim gerektirmektedir. Bu mimarinin verimi yüksektir (her bir çevrimde bir aday konum için MFT hesaplanabilir).

[84]'de Yang, Sun ve Wu ilk HK yongasını gerçekleştirmişlerdir. Bu çalışmada veri yayınlama tekniği kullanılarak iki adet seri girişli ancak paralel işlem yapabilen bir boyutlu (1B) sistolik dizi ile HK işlemini gerçekleştiren iki donanım mimarisi önerilmiştir. Bunlardan birisi referans çerçeve verisini yayınlayıp güncel blok verisini İP'ler boyunca ilerletirken diğeri güncel blok verisini yayınlayıp referans çerçeve verisini İP'ler boyunca ilerletmektedir. Yayınlama tekniği ile bir miktar küresel (global) yönlendirme (routing) gerekirken bir saat darbesinde erişilmesi gereken bellek bit genişliği gözle görülür oranda düşürülmüştür. İP sayısı bir satırdaki arama konumu sayısına eşittir. Her bir İP sadece mutlak piksel farkını hesaplamakla kalmaz aynı zamanda kısmi MFT değerini de bir önceki İP'den gelenle toplayıp bir sonraki İP'ye gönderir. %100 yararlanım sağlayabilmek için arama konumundaki satır değişimlerinde iki adet referans çerçeve pikseli de çekilir ancak bu verinin doğu İP'lere gönderilmesini sağlamak için bir çoğullayıcıya gereksinim duyulur.

[85]'de Hsieh ve Lin, bellek bit genişliğinin azaltılmasına ve arama aralığının esnekleştirilmesine yoğunlaşmışlardır. Önerilen yapı, 2B bir sistolik dizi ve kaydırmalı saklayıcı dizisinden oluşmaktadır. Önerilen mimaride, N^2 adet İP vardır ve güncel blok piksellerinin tümü ilgili İP'de bulunmaktadır. Son satır hariç her bir İP satırının ardından $2p-2$ adet kaydırmalı saklayıcı (KS) (shift register) gelmektedir. KS kullanılmasındaki esas amaç seri veri girişi avantajından faydalanmaktır. Bir arama alanı pikseli, ilk satırdaki İP ve KS'ler boyunca ilerledikten sonra ikinci satırdaki İP ve KS'lere geçerek ilerlemesini sürdürür ve bu böyle devam eder. İP'ler kısmi MFT sonuçlarını yukarı yönde ilerletirler. Her bir saat darbesinde sadece bir

adet arama alanı pikseli girer. Tüm arama alanını taramak $(N+2p-1)^2$ saat darbesi sürer ve bu süre içerisinde MFT hesabında kullanılan faydalı saat darbesi sayısı $4p^2$'dir. İP kullanım veriminin düşük olması ve KS'lerin kapladığı alan, bellek erişimi için gerekli bit genişliği ile bir ödünleşim gerektirmektedir.

[86]'da Jehng, Chen ve Chiueh tamamen farklı bir HK yapısı veren etkin ve basit bir ağaç mimarisi önermişlerdir. Ağaç yapısı sadece TABUY'yi desteklemekle kalmamakta ayrıca arama alanını azaltarak 3 adım arama (3AA) gibi hızlı yöntemleri de desteklemektedir. Bellek değiştirme ve sıra düzen değiştirme kavramları önerilerek, desteklenen bellek bant genişliği geliştirilmiş ve donanım, boşta çalışma durumuna geçmekten kurtarılmıştır. Buna karşın blok boyutu genişletildiğinde, ağacın bit genişliği çok geniş olmaktadır. Bu nedenle hız ve bellek bant genişliği arasında ödünleşim yapabilmek için $1/2^m$-alt ağaç kümesi tanımlanmıştır. Arama alanı önbelleği ve yılan arama tekniğini uygulamak, bit genişliği problemine bir başka çözümdür. Ön belleklerin girişleri 17, çıkışları ise 256 pikseldir.

[87]'de Chang ve arkadaşları, TABUY için ölçeklenebilir sistolik dizi yapıları önerilmiştir. Her bir dizi yapısının sağladığı avantaj ve dezavantajlar incelenip farklı gereksinimler için hangi yapının uygun olduğu belirlenmiştir. Farklı yapılardaki sistolik dizilerin kullanılması ile etkin bir donanım gerçeklemesi sağlanabilmektedir.

[88]'de Yeo ve Hu, [84]'deki aday çerçeveyi $2p$ tane İP'ye yayınlayan, 1B doğrusal diziyi $2p \times 2p$ boyutlu, ızgara şeklindeki bir diziye dönüştürmüştür. Güncel blok pikselleri, her bir satırdaki son İP'nin bir sonraki satırdaki ilk İP'ye bağlı olduğu dizi boyunca ilerler. Aday çerçeve verisi sadece yatay düzlemde değil aynı zamanda düşey düzlemde de yayınlanır. Temel ızgara dizide, blok boyutu *N,* $2p$'ye eşit olmalıdır. Geniş arama aralıklarında birden fazla ızgara dizi kullanılabilir. Arama aralığı, *N*'nin katlarına bölünür ve her bir bölüt bir ızgara dizide işlenir. Yukarıda bahsedilen özelliklere sahip tipik bir mimari, önceki TABU donanımı mimarisi ile karşılaştırılınca en hızlı işlemi yaparken en düşük bellek erişimi gereksinimine sahiptir. Bunun yanında, birden fazla ızgara dizideki toplam İP sayısı göreceli olarak fazladır. Sonuç olarak bu mimarinin yukarı uçtaki (high-end) uygulamalara daha uygun olduğu söylenebilir.

[89]'da Lai ve Chen, veriyi yeniden kullanma (VYK) işlemini 2B yapmak için, 1B bir İP dizisi ve iki adet birbirine geçmeli KS dizisi önermişlerdir. Aslında bu yapı, [84]'deki, güncel blok verisinin yayınlandığı 1B doğrusal diziye bir eklenti olarak nitelendirilebilir. [88]'deki bir farklılık ise arama bölgesi bölümünün her zaman gerekli olmamasıdır.

[11]'de Yeh ve Lee, güncel bloğu içinde bulunduran (iç tip) 2B dizilerde, örtüşen veri akışı kullanıldığında İP kullanım verimliliğinin arttığını tespit etmiştir. Sınırda bir aday bölge algılandığında, arama bölgesinin iki satırına da gereksinim oluşur dolayısıyla iki giriş gereklidir. Bellek bant genişliğini düşürmek için, [85]'deki KS'lere benzer bir akış belleği kullanılmıştır. Bunun yanında, [85]'den farklı olarak iki adet arama alanı pikseli eş zamanlı yayılır. Arama konumundaki satır değişikliği esnasında boşa giden saat darbesi yoktur dolayısıyla kullanım verimliliği daha yüksektir.

[90]'da VYK işlemi dört düzeyde incelenmiştir. $N \times N$, $[-p, p-1]$, $W \times H$ ve fr sırayla, blok boyutu, arama aralığı, çerçeve boyutu ve çerçeve hızı olsun. A düzeyi, iki adet yatayda komşu aday blok arasında, örtüşen $N \times (N-1)$ tane pikselin yeniden kullanılmasıdır. B düzeyi, dikeyde komşu iki adet aday bölge bandındaki $(N+2p-1) \times (N-1)$ tane ortak pikselin yeniden kullanılmasıdır. C düzeyi, yatay olarak komşu iki MB'nin, arama bölgeleri arasındaki $(2p-1) \times (N+2p-1)$ adet ortak pikselin yeniden kullanılmasıdır. D düzeyi, dikey olarak komşu iki arama bölgesi bandındaki, $(W+2p-1) \times (2p-1)$ adet ortak pikselin yeniden kullanılmasıdır. Çok büyük ölçekli tümleştirme (ÇBÖT) (VLSI - Very Large Scale Integration) teknolojisinin getirdiği YÜ SRAM limitleri nedeniyle, C düzeyinde VYK bugün en yaygın kullanılan yöntemdir. Tümleştirme kapasitesi arttıkça, D düzeyi VYK da yapılabilir hale gelecektir. Arama alanı pikselleri, YÜ'deki bellekte tamponlanmaktadır. İmge içerisinde en soldaki MB için, $(N+2p-1) \times (N+2p-1)$ adet piksel, yonga dışındaki DRAM'den YÜ SRAM'ye yüklenmektedir. Diğer, yatayda komşu MB'ler için ise arama aralığının sadece belirli bir kısmı - $N \times (N+2p-1)$ piksel - için DRAM'den SRAM'ye veri aktarılmaktadır. Dolayısıyla veri hattı bant genişliğinde $((2p-1)/(N+2p-1))$ oranında tasarruf sağlanarak $((N+2p-1) \times (N+2p-1)) \times (W/N) \times (H/N) \times fr$ olan bellek bant genişliği $(N \times (N+2p-1) \times (W/N) \times (H/N) \times fr$'ye düşmektedir. Bu aşamada, yukarıda bahsedilen kısıtlamalardan dolayı mümkün olmayan D düzeyinde VYK ileride

yapılabilir hale geldiğinde veri hattı bant genişliği 5 kat daha düşecektir. D düzeyi VYK, güç tüketimini daha etkin hale getirebilir çünkü yonga dışındaki DRAM'den veri okumak YÜ'deki SRAM'den veri okumaya göre yaklaşık on kat daha fazla güç tüketimine neden olur.

[91]'de çoklu referans çerçeve kullanılarak HK işlemi için bir VYK yöntemi önerilmiştir. [91]'de önerilen yöntem aday çerçevedeki arama sırasını ve arama merkezini değiştirerek VYK yönteminin uygulanabilirliğini artırmaya çalışmaktadır. Bu şekilde yonga üzeri belleklerden veri okuma oranı düşürülmektedir. [91]'de, gerçekleştirilen deneylerde bellek yazma oranının önerilen yöntemle geleneksel çoklu aday çerçeve seçme yöntemlerine göre %15-%30 arasında azaltıldığı belirtilmiştir.

[92]'de yapılan çalışmada, [90]'da tanımlanan VYK düzeylerinden C düzeyi ile D düzeyi arasında VYK gerçekleştirebilen bir donanım mimarisi önerilmiştir. Bu sayede arama penceresinde yatay doğrultuda örtüşen bütün veri yeniden kullanılırken dikey doğrultuda örtüşen veriler kısmen yeniden değerlendirilebilmektedir. Bu sayede C düzeyine göre daha etkin bir bellek okuma işlemi gerçekleştirilmiş daha aşağılara çekilmiştir. 720p çözünürlükte ve [-128,128] arama aralığında C düzeyinde VYK'ye göre bellek bant genişliği %54 azaltılmış ve yonga üzeri bellek büyüklüğü de %12 azaltılmıştır.

Geçmişte önerilen, başka TABU donanımı mimarileri de bulunmaktadır. Örneğin, [93]'de, [85]'de önerilen mimari, ÇSAEY ile birleştirilmiş ve gereksiz MFT hesaplamaları giderilerek güç tüketimi azaltılmıştır. [94]'de, 1024 İP içeren, 165 *Gişlem / saniye*'lik güçlü bir yonga tasarlanmıştır. [95]'de, [72]'de önerilen 1B doğrusal dizi değiştirilerek, H.263+ standardı için yarım piksel doğrulukta HK ve ayrıca AP, PB ve RRU kipleri desteklenmiştir. [96]'da, [83]'de önerilen yılan arama yapısı, düzenli kolon arama ile değiştirilmiş, İP'ler için dairesel kaydırma yapısı uygulanmış ve KS sayısı yarıya düşürülerek mimari geliştirilmiştir.

[97]'de değişken blok boyutlu HK (DBBHK) işlemini gerçek zamanlı gerçekleştirebilen bir donanım mimarisi önerilmiştir. [98]'deki çalışmada ise [97]'de önerilen mimaride kullanılan 256 adet İP yerine 36 adet İP içeren ancak daha düşük donanım maliyeti ile yaklaşık aynı performansın elde edilmesine olanak sağlayan bir donanım mimarisi önerilmiştir. [9]'da önerilen mimari ise [86]'da önerilen MFT ağacı

mimarisini kullanarak değişken blok boyutlu HK işlemi gerçekleştiren daha güncel bir çalışmadır. [99]'da önerilen çalışmada değişken blok boyutlu HK işlemi için dolambaçlı arama yöntemini kullanan ve bu sayede verilerin daha verimli bir şekilde kullanılmasını sağlayarak güç tüketimini azaltan yeniden düzenlenebilir bir donanım mimarisi önerilmiş ve ASIC gerçeklemesi yapılmıştır.

Yapılan araştırmalara bakıldığında sistolik dizilerin TABU donanımı mimarileri için tercih edilen yaklaşım olduğu açıktır. Yonga alanı ve çıktı miktarı, MFT gecikmesi ve bellek bant genişliği, İP kullanım verimliliği ve veri hizalama devreleri, veri hattı genişliği ve YÜ SRAM, birbirleri ile ödünleşim halinde olan değişkenlerdir. Tasarımcıların bu noktada hedef uygulama için feda edilebilecekleri ve olmazsa olmaz değişkenleri çok dikkatli belirlemeleri gerekmektedir.

3.2.1 Hızlı Blok Uyumlama Mimarileri

Hızlı HK yöntemleri video kalitesini kabul edilebilir bir seviyede tutarken TABU'nun gerektirdiği yüksek işlem sayısını çok aşağılara çekebilirler. Hızlı HK mimarilerinin tasarımında en çok çaba gerektiren adımlar, öngörülemez veri akışı, düzensiz bellek erişimi, sistolik dizilere haritalama aşamasındaki zorluklar, donanım kullanım verimliliğinin az oluşu ve paralelleştirilemeyen veri bağımlılığı içeren ardışık yordamlardır. Hızlı HK mimarilerinin silikon üzerinde kapladıkları alan TABU donanımı mimarilerine oranla nispeten küçük olmaktadır [14].

[100]'de Jong ve arkadaşları, 3AA yöntemi için tamamen ardışık düzende çalışabilen bir donanım mimarisi geliştirmişlerdir. Temel olarak dokuz adet İP, her adımda dokuz farklı aday bölgenin MFT'sini hesaplamaktadır ve üç adımın her biri 256 saat darbesinde tamamlanmaktadır. 3AA'nın bütün avantajlarından yararlanabilmek için, veri düzenlemesi ve bellek düzenlemesi uygun şekilde yapılmıştır. Önerilen mimarideki sistolik dizi, MB'leri ardışık düzene koyarken, çıktı miktarını üçe katlayarak 27 İP içerecek şekilde genişletilebilmektedir. 27 İP'li tasarımın gecikmesi, 9 İP'li tasarımla aynıdır. Önerilen mimarinin 3 İP ile tasarlanan hali kullanılarak yonga alanı küçültülebilir. Böylece çıktı miktarı üç kat azalırken, gecikme de üç kat artacaktır.

Her üç tasarımında %100'e yakın bir kullanım verimliliği vardır. 256 İP içeren bir TABU donanımı mimarisi ile 9 İP içeren bir mimarideki toplam kapı sayısı karşılaştırıldığında ikinci mimarinin içerdiği kapı sayısı daha düşüktür (36.6K'ye 192.2K). 27 adet İP kullanıldığında, çıktı oranı neredeyse TABUY ile aynı olmaktadır ancak kapı sayısı 110K civarında olup yine daha azdır.

[101]'de Dutta ve Wolf, [84]'de önerilen mimarinin veri akışını değiştirerek; TABUY, 3AA ve türemiş doğrultuda aramayı (conjugate direction search) aynı mimaride destekleyen bir yapı geliştirmişlerdir. Çok katlı bir ara bağlantı şebekesi aracılığıyla İP'lerle haberleşebilen çoklu bellek bankaları düzenlenmiştir. 3AA, hedef HK yöntemi olarak seçildiğinde çıktı oranı TABUY'ye göre sekiz kat daha fazla olmaktadır. 1B dizinin programlanabilir olması, onu farklı zamanlama ve güç kısıtlamaları içeren uygulamalar için de kullanılabilir hale getirmektedir.

[102]'de Lin ve arkadaşları, programlanabilir bir HK yongası için, algoritma ve mimarinin eş zamanlı tasarlanmasını önermişlerdir. YÜ'de etkin bir şekilde yürütülebilen, makro komutlar şeklindeki arama yöntemleri kullanılarak değişik yöntemler gerçeklenmiştir. İki farklı programlanabilir mekanizma desteklenmektedir. Birisi alt örneklenmiş arama konumu ve/veya blok pikselleri diğeri de küme aramadır. Sabit arama örüntülerinin yığın (batch) şeklinde yürütülmesine karşın programlanabilir HK'de makro-komutlar, etkileşimli bir biçimde yürütülmektedir. Dolayısıyla MFT'nin hesaplanması için geçen süre düşük olmalıdır. Bu durumda, seri dizi mimarisi uygun değildir, bu nedenle paralel, 2B dizi seçilmiştir. Geleneksel olarak SRAM ile 2B paralel dizi arasında veri hizalamasını sağlayabilmek için birden fazla SRAM bankası kullanılmaktadır. Bu çalışmada, karmaşık ara-yüze gerek duymayan kendinden hizalanabilen piksel döndüren İP'ler ile çift adreslemeli tek bağlantı noktalı bir bellek kullanılmıştır. Bir ilk örnek yonga gerçeklenmiştir. Elde edilebilen en yüksek hesaplama kapasitesi 14 *Gişlem / saniye'*dir ve bu miktar 30 çerçeve/s'de CIF çözünürlük için yeterlidir.

[103]'de Cheng ve Hang, birçok hızlı blok uyumlama yöntemini (HBUY) gerçekleyen, genel bir sistolik dizi mimarisi kullanmışlardır. Özetle; yonga alanı ve giriş çıkış bant genişliği kullanım verimliliğinin, büyük oranda çerçeve boyutu ve arama aralığına bağlı olduğunu görmüşlerdir. Küçük video çerçeveleri ve yavaş hareket olan durumlarda, inceledikleri bütün HBUY'ler benzer başarım göstermiştir. Geniş imge ve hızlı hareket olan durumlarda ise, belli hızlı yöntemler, gözle görülür oranda

düşük yonga alanına kaplamaktadırlar. Aslında belli başlı bazı HBUY'ler için sistolik diziler yerine [100]'de, 3AA için geliştirilen mimaride olduğu gibi daha etkin, adanmış donanım tasarımı yapılırsa hızlı yöntemlerinin kapladıkları yonga alanı daha az olabilir. Bu, alt seviye uygulamalar için daha uygun bir çözümdür.

[104]'de Minzuno ve arkadaşları, ardışık ve paralel gerçekleştirilen geleneksel HK tasarımlarında, arama penceresi boyutuna, tasarım bu değişkene bağlı olarak eniyilendiği için ilk başta karar verilmesi gerektiğini görmüşlerdir. Arama penceresi boyu değiştiğinde, donanım mimarisi yeterli etkinliği gösterememektedir. Bu durum göz önünde bulundurarak, arama penceresi boyutu değişiminden etkilenmeyen bir HK ve gerçekleme yöntemi geliştirmişlerdir. İki adımlı sıradüzensel arama yapısını kullanmışlardır. İlk adımda hareket vektörü, yatayda iki piksel ve dikeyde bir piksel doğrulukta kabaca bulunmaktadır. İkinci adımda ise ilk adımda bulunan nokta etrafındaki 5×3 noktada hassas arama yapılmaktadır. Nihai hareket vektörleri yarım piksel doğrulukta elde edilmektedir. Fazla detay içeren video dizilerindeki kalite kaybı yaklaşık $0.13dB$ civarındadır. Tipik video dizilerinde ise bu rakam yaklaşık $0.03dB$'dir. HK arama aralığı, yatayda $-48/+47$ ve dikeyde $-16/+15.5$ 'dir. Arama penceresi, uyarlanabilir olarak kaydırılarak elde edilen yalancı (pseudo) arama aralığı, yatayda $-96/+95$, dikeyde $-32/+31.5$'dir ve kodlama etkinliğini $0.4/0.8dB$ civarında iyileştirir. [62, 105]'de sıra düzensel HK yapan diğer mimarilere örnekler bulunmaktadır. Bu çalışmalarda 3 seviyeli sıra düzensel yapı kullanılmıştır. [104]'de, farklı seviyelere ait işlemler farklı işlem birimlerinde gerçekleştirilir. Dolayısıyla farklı sevilerdeki işler için MB'ler ardışık düzende işlenebilir. Diğer taraftan [62]'deki mimari, yonga alanından tasarruf etmek için farklı seviyelerdeki HK için temel işlem birimini ardı ardına kullanır.

[106]'da Moshnyaga, yeni bir hızlı yöntem geliştirmiş ve bu yöntemde arama penceresi boyutunun güncel blok ve aday blok arasındaki farkın düzeyine göre kademeli olarak azaltılabileceğini önermiştir. i. satır sonunda elde edilen kısmi MFT değeri belli bir eşikten büyükse, takip eden adımlar için gerekli arama işlemleri tamamen atlanabilir. Uyarlanabilir yöntemi uygulamak için, [84]'deki 1B doğrusal dizi ve AB2 [10] örnek olarak seçilmiştir. Buna karşın sadece güç tüketimi azaltılabilmiştir. Çünkü zaten belirlenmiş olan veri akışı yapısı nedeniyle yonga alanında ve bir MB için geçen gecikme zamanında bir iyileşme olamamıştır. Eşik değiştirme yöntemi ile yüksek kalite korunurken %75'e yakın hesap tasarrufu sağlanabileceği ifade edilmektedir.

[107]'de Hsia, hareket vektörünün zamansal olarak öngörülmesini ve gözle görülür derecede küçük bir arama penceresi kullanarak düzeltilmesini önermiştir. Sadece sekiz İP'den oluşan ve 8k adet kapı kullanılarak yapılan tasarım, $53kMB/saniye$'lik bir etkinlik sağlamıştır. Üretilen hareket vektörleri $[-127,+127]$ aralığında olabilir. Zamansal hareket vektörü öngörücünün etrafında, $-7/+7$ aralığında uyarlanabilir tam arama kullanılmıştır. Aslında geleneksel tam aramaya kıyasla bu yöntem kullanıldığında karşılaşılan kayıp çok büyük olabilir. Örneğin *"football"* dizisinde çok hızlı kamera hareketleri vardır. Bu gibi senaryolar için, bu yöntem PSNR değerlerinde yaklaşık $2dB$ kayba neden olabilmektedir.

[108]'da Kawahito, HK işlemini CMOS algılayıcılarla birleştirmiştir. Çerçevelerin algılayıcıdan yüksek hızla gelmesinden faydalanılarak; uyarlanabilir yinelemeli-arama temelli BUY önerilmiştir. Aslında arkadaki mantık son derece basittir. Ara çerçeve hızı çok yüksek olduğu için daha düşük çerçeve hızlarında, nispeten büyük arama pencerelerinde gerçekleştirilen HK işlemi bu hızlı video çerçeveleri üzerinde çok daha düşük hızlarda gerçekleştirilir. Ardından, elde edilen hareket vektörleri istenilen çerçeve oranına göre birleştirilebilir. Örneğin çerçeve hızı ve arama aralığı sırasıyla 30 ve $[-64,+63]$ olsun. Eğer çerçeve oranı 480'e çıkarsa arama aralığı $[-4,+3]$'e düşecektir. Aynı BUY'nin kullanıldığı düşünülürse, gereken hesaplama sayısının oranı $128^2 \times 30 \div 8^2 \times 480 = 16 \div 1$ olacaktır. Azalan karmaşaya ek olarak arama aralığı küçüldüğü için bellek erişimi de azalacaktır.

[109]'da Vleeschouwer ve arkadaşları, doğrultuya bağlı karelenmiş (directional squared-search) BUY'yi önermişlerdir. Diğer; güncel, hızlı BUY'lere göre benzer başarıma sahiptir. Çünkü hareket vektörlerinin merkez eğilimli olması, belli bir eşik yakalandığında arama işleminin erken sonlandırılması, en uyguna daha hızlı yakınsama gibi düşünceler ortaktır. Bununla birlikte, yöntemin tasarımında, komşu aday bloklardaki verilerin değerlendirilmesi gibi mimari endişeler gözetilmiştir. Yatay ve dikeyde komşu üç aday blok için üç İP kullanılmıştır. Deneysel sonuçlara bakıldığında elmas aramaya kıyasla HK başarımı bir miktar düşük, donanım karmaşası da fazladır. Ancak önerilen yöntem daha az bellek erişimi yaptığı için önerilen bu mimari daha etkindir. Herhangi bir benzetim sonucu veya donanım gerçeklemesine ait veri sunulmamıştır.

[110]'da Chao ve arkadaşları, melez bir HK mimarisi gerçekleştirmişlerdir. Önerdikleri mimari, bölgesel elmas arama [27] ve hızlı tam arama [64] yöntemini desteklemektedir. C düzeyinde VYK yapabilen donanım, CIF çözünürlükte $30\ çerçeve/saniye$ ve $[-16,+15]$ arama aralığında çalışabilmektedir. Arama alanı pikselleri YÜ bellekte tamponlanmaktadır. Farklı kolonlardaki arama alanı pikselleri, farklı bellek banklarına paylaştırılmıştır. Böylece arama alanındaki 8×1'lik piksel gurubuna aynı saat darbesinde erişmek mümkündür. MFT'yi 32 saat darbesinde hesaplayabilmek için 8 İP'lik bir MFT ağacı [86] kullanılmıştır. Elmas arama kipi seçildiğinde bazı aday bölgeler kontrolsüz bir şekilde birden fazla kez aranmaktadır. Hesap artıklığını önlemek amacıyla 1024 adet bir-bitlik bayrak kullanılmakta ve bir konumun zaten incelenip incelenmediği bu bayraklar yoluyla denetlenmektedir. Elde edilen hesaplama kazancı, ortalama $\%23.43$ civarındadır. Bu çalışmada yazarlar, bayrakları ortadan kaldırmış ve bir sonraki arama konumunun belirlenmesi için ROM tabanlı bir yöntem kullanmışlardır. Bu şekilde elde ettikleri hesaplama kazancı da yaklaşık $\%23.23$ olmuştur. Bu, geleneksel yöntemle elde edilen tasarrufa çok yakındır. Toplanan MFT'yi belli bir MFT_{min} değeri ile karşılaştırmak ve hesaplama zamanını azaltmak için KBE yöntemi de kullanılmıştır. Tasarım AEY kipi için ayarlandığında bir aday bölge için MFT hesabının gerekli olup olmadığına karar verecek fazladan devreler etkinleştirilmektedir. Her bir arama konumu için karar verme işlemi bir saat darbesi sürmektedir. Ancak; eğer atlama koşulu sağlanmazsa, MFT'yi hesaplamak için 32 saat darbesi gerekmekte ve MFT karar devresi boşuna çalışmaktadır. Bu nedenle veri çıktısını artırmak amacıyla 8-İP MFT ağacı ile MFT karar devresi arasına bir FIFO bellek yerleştirilmiştir. Bu mimari *MPEG-4, simple profile, level 3* kodlayıcısının VLSI tasarımı ile başarılı bir biçimde tümleştirilmiştir ve HK donanımının kapı sayısı sadece 9k'dır. [110]'da, bu mimarinin yarım/çeyrek piksel doğrulukta çalışan hesaplama kontrollü bir yapıya eklenmiş hali bulunabilir.

Yapılan araştırmalara bakıldığında hızlı BUY donanımı mimarilerinin tasarımında eğilim, yöntem ve mimari tasarımın eş zamanlı olarak gerçekleştirilmesi yönündedir. Algoritma seviyesindeki geliştirmelerin getirisi mimari seviyede yapılan geliştirmelere göre daha yüksek görünmektedir. Sadece yakınsama hızı ve yerel minimumlara takılmaktan kaçınmak gibi geleneksel, yöntem sıkıntıları değil, mimari meseleler de göz önünde bulundurulmalıdır. Örneğin; rastgele sekiz aday bölgeyi aramak için gerekli bellek okuması sayısı $16\times16\times8=2048$ piksel iken, birbirine komşu sekiz aday bölgeyi aramak için bu rakam $16\times(16+7)=368$ pikseldir.

Bu durumda, çizgisel aramanın bellek erişiminde daha etkin olduğu ortaya çıkmaktadır. Dolayısıyla TABUY için seri sistolik diziler uygun çözüm gibi görünseler de bir sonraki aday konumun belli olmadığı, hızlı BUY'lerde veri çıktısını (data throughput) düşürebilirler. Çünkü dizideki bütün İP'lerin doldurulması için uzunca bir süre gerekmektedir. Dolayısıyla doldurulmaları için daha geniş bellek bant genişliği gereken, paralel yüklemeli mimariler hızlı BUY'ler için ideal çözüm gibi durmaktadırlar.

3.3 İP'lerin Çalışma Şekline Göre Donanım Mimarileri

Geçmişte önerilen mimarilere, içerdikleri İP'lerin çalışma şekli esas alınarak bakılacak olunursa iki mimarinin kullanıldığı görülecektir. Bunlardan birisi kaynak piksel temelli doğrusal dizi (KPTDD) mimarisi diğeri de hareket vektörü temelli doğrusal dizi mimarisidir (HVTDD) [55].

[48, 55]'de, geçmişte önerilen, düşük bit derinliğinde HK yapan mimariler görülebilir. [48]'de HVTDD kullanılarak tasarlanmış 1BD temelli bir HK mimarisi görülmektedir. KPTDD ise [55]'de önerilmiş ancak düşük bit derinliğindeki HK yöntemlerine uygulandığı literatürde görülmemiştir.

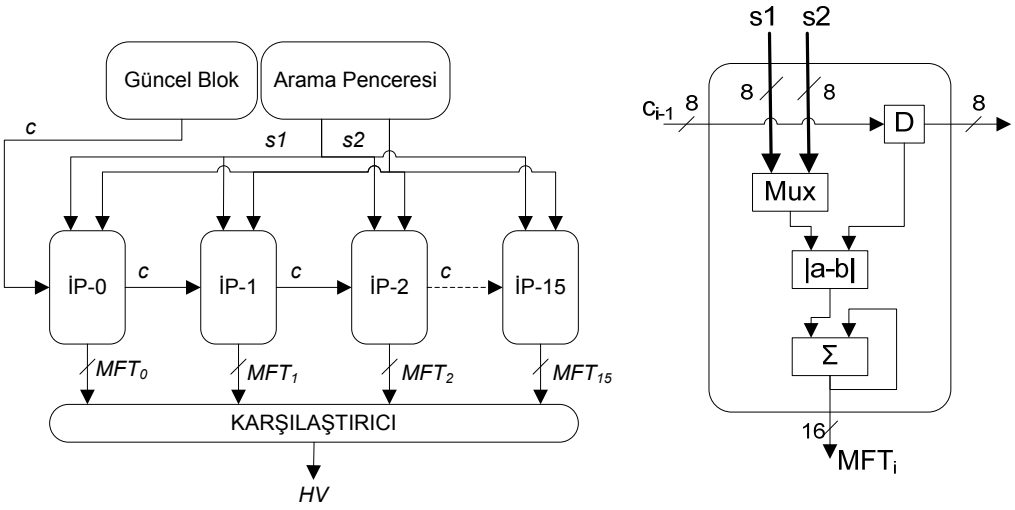
[49]'da sıradüzensel arama yöntemi ikili piramit yapısı kullanılarak düşük bit derinliğinde HK gerçekleştiren bir mimari önerilmiştir. Bu arama yönteminin, Ç1BD yöntemine göre hesapsal karmaşası daha fazladır. Çünkü üç farklı bit düzlemi için bellek erişimine gereksinim duyulmaktadır. Bu çalışmada herhangi bir ASIC veya FPGA gerçeklemesine yer verilmemiş, yöntemin sadece standart mikroişlemciler üzerindeki çalışması incelenmiştir

İkili doğası nedeniyle, düşük bit gösterimi temelli HK yöntemleri ve donanım mimarileri, geçmişte bolca önerilmiş olan 8-bit gösterimi temelli mimarilere göre, gerek düşük hesap yükü gerekse kapladıkları yonga alanı bakımından daha avantajlıdırlar. 8-bit gösterimi temelli mimariler, gerçek zamanlı video kodlama işlemlerinde kullanılabilir olmaları için 2B sistolik diziler kullanılarak tasarlanırlar. Bu durum sahip oldukları donanım karmaşasını daha da artırmaktadır.

Düşük bit derinliği temelli HK mimarilerinde, yalnızca 1B sistolik dizi kullanarak gerçek zamanlı video kodlama mümkün olabilmektedir. [55]'de, [48]'de önerilen mimarinin, H.263 kodlayıcı üzerindeki performansının geleneksel MFT ölçütü ile kıyaslanabilir olduğu açıkça ortaya konulmuştur.

3.3.1 HVTDD mimarisi

HVTDD mimarisinde, her bir İP, bir arama konumu için gerekli uyum ölçütünün hesaplanmasından sorumludur. Bu donanım mimarisinin ve içerisinde kullanılan İP'nin öbek gösterimi sırasıyla, Şekil 3.2(a) ve Şekil 3.2(b)'de görülmektedir.



Şekil 3.2: HVTDD mimarisi a) Öbek gösterimi b) İP öbeğinin iç yapısı

Şekil 3.2(a)'da görülen dizideki her bir İP, ayrı bir uyum ölçütü değerini hesaplayabilir ve 16 farklı uyum ölçütü değeri paralel bir biçimde hesaplanabilir. 256 farklı uyum ölçütü değerini hesaplamak için güncel blok 16 kez ardı ardına taranmalıdır. Tablo 3.1'de bu mimaride kullanılan veri akışı yapısı görülmektedir.

Tablo 3.1'de görüldüğü üzere *c* ile gösterilen güncel blok verisi girişinden gelen veri her bir çevrimde bir sonraki işlem öbeğine aktarılmaktadır. *s1* ve *s2* ile arama penceresi verisi girişlerinden gelen veri, her bir saat darbesinde değişmektedir ve İP içinde arama konumuna/noktasına bağlı olarak yalnızca biri kullanılmaktadır. Şekil 3.2(a)'ya bakıldığında bu durum daha net anlaşılabilir. Güncel blok verisini içeren bellek öbeğinden tek bir çıkış alınmaktadır Buradan gelen veri her bir çevrimde birinci İP öbeğine girer ve bir önceki çevrimde bu öbeğe giren veri bir sonraki İP

öbeğine gider. Sonuçta her bir İP öbeği farklı güncel blok verisi içerdiği için 16 farklı mutlak fark değeri bir saat darbesinde hesaplanabilir. Tablo 3.1'e bakıldığında tüm İP öbeklerinin çalışır duruma gelmesi için başlangıçtan itibaren 15 saat darbesi gerektiği görülür. Dolayısıyla bir güncel blok için hareket vektörünün bulunmasında gereken saat darbesi sayısı olması gerekenden 15 fazla olacaktır.

[84]'de, bu mimari hakkında daha detaylı bilgiye ulaşılabilir.

Tablo 3.1: 16×16 blok boyutu ve $[-8,+7]$ arama aralığı için HVTDD mimarisinde kullanılan veri akışı yapısı

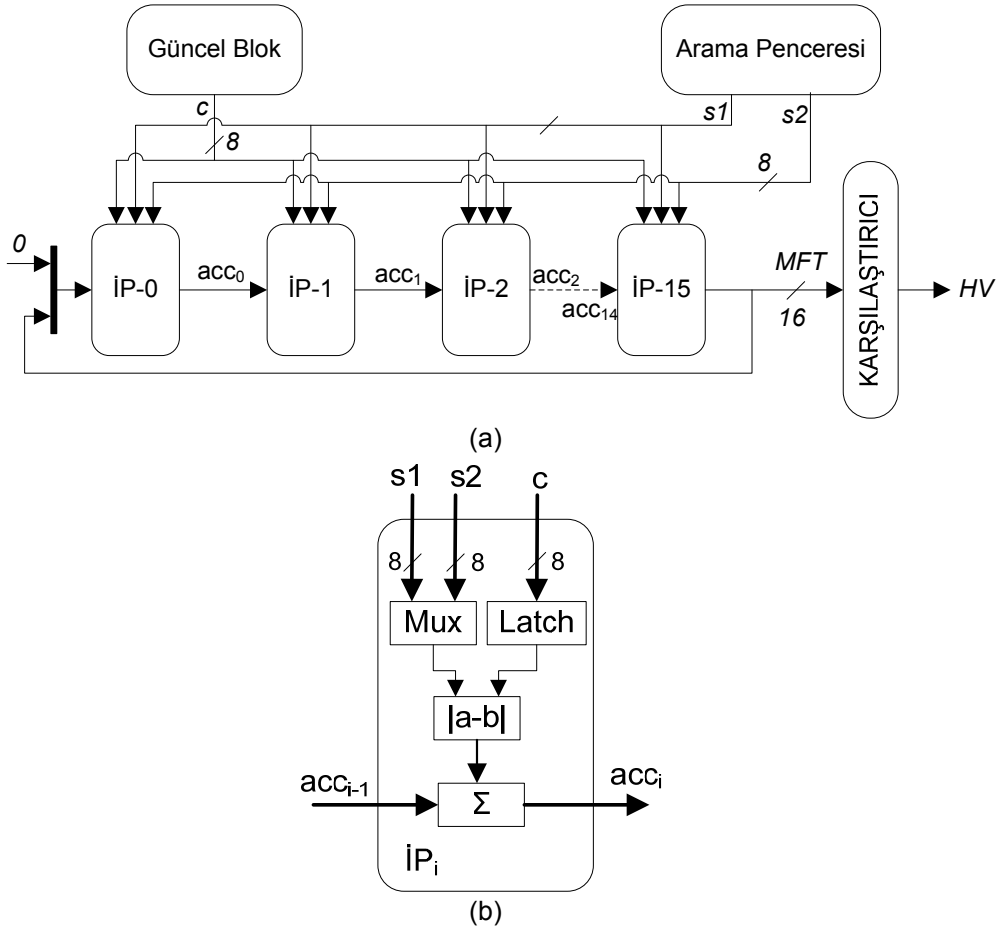
Çevrim zamanı	Giriş Verisi			İşlemci Girişleci			
	c	*s1*	*s2*	**İP-0**	**İP-1**	…	**İP-15**
0	*c0,0*	*s0,0*		*c0,0-s0,0*			
1	*c0,1*	*s0,1*		*c0,1-s0,1*	*c0,0-s0,1*		
2	*c0,2*	*s0,2*		*c0,2-s0,2*	*c0,1-s0,2*		
⋮		⋮		⋮	⋮	…	⋮
14	*c0,14*	*s0,14*		*c0,14-s0,14*	*c0,13-s0,14*		
15	*c0,15*	*s0,15*		*c0,15-s0,15*	*c0,14-s0,15*		*c0,0-s0,15*
16	*c1,0*	*s1,0*	*s0,16*	*c1,0-s1,0*	*c0,15-s0,16*		*c0,1-s0,16*
17	*c1,1*	*s1,1*	*s0,17*	*c1,1-s1,1*	*c1,0-s1,1*		*c0,2-s0,17*
⋮		⋮		⋮	⋮	…	⋮
30	*c1,14*	*s1,14*	*s0,30*	*c1,14-s1,14*	*c1,13-s1,14*		*c0,15-s0,30*
31	*c1,15*	*s1,15*	*s0,31*	*c1,15-s1,15*	*c1,14-s1,15*		*c1,0-s1,15*
⋮		⋮		⋮	⋮	…	⋮
240	*c15,0*	*s15,0*	*s14,16*	*c15,0-s15,0*	*c14,15-s14,16*		*c14,1-s14-16*
⋮		⋮		⋮	⋮	…	⋮
255	*c15,15*	*s15,15*	*s14,31*	*c15,15-s15,15*	*c15,14-s15,15*		*c15,0-s15,15*
256			*s15,16*		*c15,15-s15,16*		*c15,1-s15,16*
257			*s15,17*				*c15,2-s15,17*
⋮			⋮	⋮	⋮	…	⋮
270			*s15,31*				*c15,15-s15,30*

3.3.2 KPTDD mimarisi

KPTDD mimarisinde, HVTDD mimarisinden farklı olarak; her İP, bir aday bölge için uyum ölçütünün hesaplanmasından değil de güncel blok üzerindeki belirli bir piksel/pikseller ile aday bloktaki ilgili piksel/pikseller arasındaki mutlak farkın hesaplanmasından sorumludur. Bu mimaride güncel blok pikselleri için bellekten yapılan okuma sayısı daha düşüktür dolayısıyla bellek bant genişliğinin az olması için, tercih edilmesi gereken bir mimaridir.

HVTDD mimarisinin KPTDD mimarisine göre dezavantajlarından biri de; karşılaştırıcı öbeği ile İP öbekleri arasında çok geniş bir veri yolu bulunmasıdır. Bu durum, güç tüketimini ve kontrol yapısını etkilemektedir.

Şekil 3.3(a)'da görüldüğü üzere bir İP, her saat darbesinde iki piksel arasındaki mutlak farkı hesaplayıp bir önceki İP öbeğinden gelen mutlak fark ile toplamaktadır. Dizideki son İP öbeği, toplam mutlak fark değerini hesaplamaktadır.



Şekil 3.3: KPTDD mimarisi a) Öbek gösterimi b) İP öbeğinin iç yapısı

Dolayısıyla KPTDD mimarisindeki İP öbekleri, HVTDD mimarisine göre birbirlerine daha bağımlı bir yapıya sahiptirler.

Şekil 3.2(b)'de görülen İP öbeği ile Şekil 3.3(b)'de görülen İP öbeği arasındaki en farklılık, güncel blok piksellerin *D* yaz-boz (flip-flop) yerine *Latch* öbeğine girmesidir. Bu durumda bellekten okuma oranının düşeceği açıktır. Çünkü bu yapıda belirli bir çevrim boyunca İP öbeğine giden *c* verisi sabit kalmaktadır. Ayrıca Şekil 3.2(b)'de görülen geri besleme yapısı Şekil 3.3(b)'deki İP öbeğinde bulunmamaktadır çünkü

bir önceki İP den gelen mutlak fark toplamı değeri İP içinde hesaplanan mutlak fark değeri ile toplanarak bir sonraki İP ye gönderilmektedir. [55]'de daha ayrıntılı bilgiye ulaşılabilir.

Tablo 3.2: 16×16 blok boyutu ve $[-8,+7]$ arama aralığı için KPTDD mimarisinde kullanılan veri akışı yapısı

Çevrim zamanı	**Giriş Verisi**			**İşlemci Girişleri**			
	c	*s1*	*s2*	**İP-0**	**İP-1**	…	**İP-15**
0	*c0,0*	*s0,0*		*c0,0-s0,0*			
1	*c0,1*	*s0,1*		*[c0,0]-s0,1*	*c0,1-s0,1*		
2	*c0,2*	*s0,2*		*[c0,0]-s0,2*	*[c0,1]-s0,2*		
.		.		.	.	...	.
.		.		.	.		.
.		.		.	.		.
14	*c0,14*	*s0,14*		*[c0,0]-s0,14*	*[c0,1]-s0,14*		
15	*c0,15*	*s0,15*		*[c0,0]-s0,15*	*[c0,1]-s0,15*		*c0,15-s0,15*
16	*c1,0*	*s1,0*	*s0,16*	*c1,0-s1,0*	*[c0,1]-s0,16*		*[c0,15]-s0,16*
17	*c1,1*	*s1,1*	*s0,17*	*[c1,0]-s1,1*	*c1,1-s1,1*		*[c0,15]-s0,17*
.		.		.	.		.
.		.		.	.	...	.
.		.		.	.		.
30	*c1,14*	*s1,14*	*s0,30*	*[c1,0]-s1,14*	*[c1,1]-s1,14*		*[c0,15]-s0,30*
31	*c1,15*	*s1,15*	*s0,31*	*[c1,0]-s1,15*	*[c1,1]-s1,15*		*c1,15-s1,15*
.		.		.	.		.
.		.		.	.	...	.
240	*c15,0*	*s15,0*	*s14,16*	*c15,0-s15,0*	*[c14,1]-s14,16*		*c14,15-s14-16*
.		.		.	.		.
.		.		.	.	...	.
.		.		.	.		.
255	*c15,15*	*s15,15*	*s14,31*	*[c15,0]-s15,15*	*[c15,1]-s15,15*		*c15,15-s15,15*
256			*s15,16*		*[c15,1]-s15,16*		*[c15,15]-s15,16*
257			*s15,17*				*[c15,15]-s15,17*
.			.	.	.	...	.
.			.	.	.		.
.			.	.	.		.
270			*s15,31*				*[c15,15]-s15,30*

KPTDD mimarisinde 8-bit/piksel gösterimi ve 1B sistolik dizi için kullanılacak olan veri akışı yapısı Tablo 3.2'de görülmektedir.

Tablo 3.2'de görüldüğü gibi her bir İP'ye güncel bloğun belli pikselleri gönderilir. Örneğin burada görülen veri akışı yapısında İP-0'a, c(0,0) ile c(15,0) arasındaki birinci sütun piksellerinin gönderildiği görülmektedir.

4. Ç1BD VE K-1BD TEMELLİ HK YÖNTEMİ VE DONANIM MİMARİSİ

Bu bölümde Ç1BD ve K-1BD temelli HK yöntemleri anlatılarak, bu yöntemleri gerçekleştirmek için bu tez çalışması kapsamında tasarlanan donanımsal mimariler tanıtılmaktadır. HK işleminin karmaşasını azaltmak için kullanılabilecek yaklaşımlardan birisi, pikselleri göstermek için daha düşük bit derinliğinin kullanılması olup Ç1BD ve K-1BD temelli HK yöntemleri bu sınıfta ele alınabilecek iki farklı HK yöntemidir.

4.1 Ç1BD Temelli HK Yöntemi ve Donanım Mimarisi

[51]'de önerilen Ç1BD, [48]'de önerilen 1BD temelli hareket kestirimi mantığını kullanmakla birlikte, bit düzlemlerinin elde edilmesi için kullanılan süzgeç yapısının değiştirilmesi ile dönüşümdeki işlem yükü azaltılmış bir HK yöntemidir.

4.1.1 Ç1BD temelli HK yöntemi

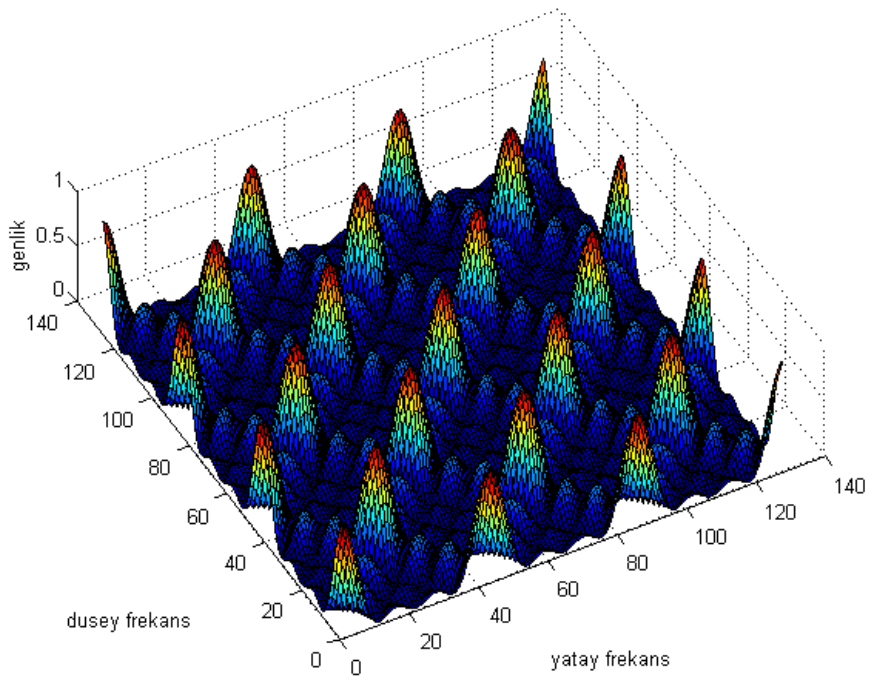
[48]'de önerilen 1BT dönüşümü basit ve nispeten iyi bir başarıma sahiptir. Ancak bu yöntemin bir dezavantajı, (4.1)'de verilen süzgeç çekirdeğini normalize ederken kullanılan kayan noktalı çarpma işlemidir.

$$K(i,j) = \begin{cases} 1/25, & eğer\ i,j \in [0,4,8,12,16] \\ 0, & diğer \end{cases} \tag{4.1}$$

Bu işlem, hem donanım hem de yazılım için nispeten yavaş gerçekleştirilebilmektedir [111]. [51]'de önerilen yöntem ile [48]'deki yöntemin HK başarımı, çekirdek normalizasyonunundaki yüksek hesap yükü olmadan elde edilmiştir. [51]'de bu durumu ortadan kaldırmak için çekirdek yapısı, süzgecin HK başarımı etkilenmeyecek şekilde değiştirilmiş ve ikinin kuvveti kadar 1 (bir) içerecek şekilde yeniden düzenlenmiştir. Bunun sonucunda normalizasyon işlemi karmaşık çarpma işlemi yerine basit kaydırma işlemi şeklinde gerçekleştirilebilir hale getirilmiştir. Bu yeni süzgeç çekirdeği (4.2)'de görülmektedir.

$$K = \frac{1}{16}\begin{bmatrix}
0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&1&0&0&0&0&0&1&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&0&0&0&0&0&1&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&1&0&0&0&0&0&1&0&0&0&0&0&1\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&0&0&0&0&0&1&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&1&0&0&0&0&0&1&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0
\end{bmatrix} \tag{4.2}$$

Buradaki çekirdek, biçimsel olarak (4.1)'de verilen süzgeç çekirdeğinin merkez etrafında 45° döndürülmüş bir şekli olarak düşünülebilir. Şekli itibarı ile [51]'de bu çekirdek elmas şekilli olarak nitelendirilmiştir. Şekil 4.1'de frekans yanıtı görülen süzgeç çekirdeği, çoklu bant geçiren bir yapı göstermektedir. Ancak (4.1)'deki çekirdeğe göre hesap yükü çok daha azdır. Dolayısıyla donanıma getireceği yük daha düşüktür.



Şekil 4.1: [51]'de önerilen çoklu bant geçiren süzgeç çekirdeğinin frekans yanıtı

Her bir video çerçevesinin tek bir bit ile gösterilmesi için gerçekleştirilen işlemler Bölüm 2.2.3.1'de gösterilmişti. Aday blok ile güncel blok arasındaki uyumu belirtmek için kullanılan ölçüte, 1BD yönteminde uyumsuz nokta sayısı (UNS) adı verilmekte olup bu ölçüt (4.3)'de gösterilmektedir.

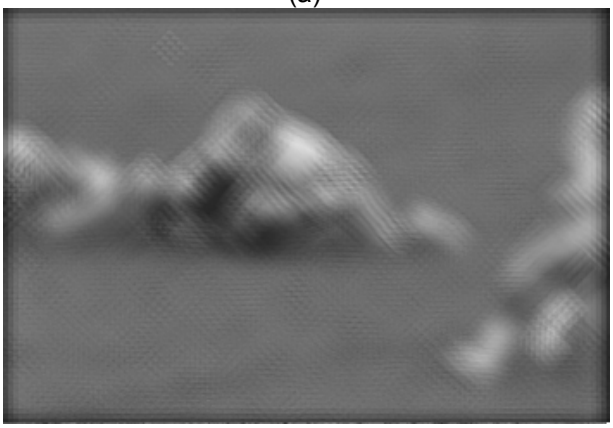
$$UNS(m,n) = \sum_{i=0}^{M-1} \sum_{j=0}^{M-1} B^{t}(i,j) \oplus B^{t-1}(i+m, j+n) \qquad -s \le m,n \le s-1 \tag{4.3}$$

Bu eşitlikteki $B^{t}(i,j)$, *t* anındaki 1BD video çerçevesini, (m,n) aday bloğun yer değiştirme miktarını, s arama aralığını, $\oplus$ XOR işlemini göstermektedir. Hareket vektörü, $UNS(m,n)$ ölçütünün en küçük olduğu aday konumdaki (m,n) değerine eşittir. Eğer herhangi iki aday bölge için elde edilen UNS değeri birbirine eşit olursa güncel blok konumuna yani *m*=*n*=0'a en yakın (en küçük Öklid mesafesindeki) hareket vektörü kullanılmaktadır. Bu durumun matematiksel ifadesi (4.4)'de verilmiştir.

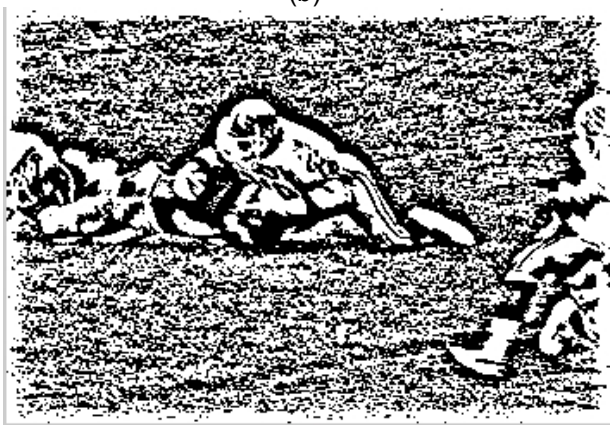
$$HV = \left\{ (u,v) \middle| UNS(u,v) \le UNS(m,n); -p \le m,n \le p-1 \right\} \tag{4.4}$$

Şekil 4.2'de *"football"* dizisinden bir çerçeve ve bu çerçeve üzerinde sırasıyla, (4.2)'deki süzgeç çekirdeği ile süzgeçlenmiş ve 1BD sonucunda elde edilen imge çerçeveleri görülmektedir.

(a)



(b)



(c)

Şekil 4.2: a) *"football"* dizisinden örnek bir çerçeve, b) Süzgeçlenmiş hali, c) 1BD sonrası.

[51]'de önerilen yöntemin başarımı farklı test dizileri üzerinde denenmiştir. Uyum ölçütü olarak sırayla OMF, [50]'de önerilen 2BD, ve [48]'deki 1BD yöntemi ve son olarak [51]'de önerilen Ç1BD kullanılmıştır. HK başarımını değerlendirebilmek için tepe işaret gürültü oranı (PSNR) ölçütü kullanılmıştır.

Tablo 4.1: Ç1BD yöntemi kullanılarak 16×16 blok boyutu için yapılan hareket kestirimi işlemi ardından yeniden oluşturulan imgeler ile asıl imgeler arasındaki farka bağlı hesaplanan PSNR(dB) değerleri ve diğer yöntemlerle karşılaştırılması [51]

Yöntem	Video dizileri(Çerçeve boyutu, dizi uzunluğu)					
	"football" (352×288) $(125\ çer.)$	*"foreman"* (352×288) $(300\ çer.)$	*"tennis"* (352×240) $(112\ çer.)$	*"flowergarden"* (352×240) $(115\ çer.)$	*"mobile"* (352×240) $(140\ çer.)$	*"coastguard"* (352×288) $(300\ çer.)$
MFT	22.88	32.10	29.87	23.79	22.99	30.27
2BD[50]	22.08	30.71	28.89	23.43	22.72	29.93
1BD[48]	21.83	30.36	28.77	23.32	22.71	29.84
Ç1BD[51]	21.81	30.39	28.78	23.29	22.73	29.88

Tablo 4.1'de farklı video dizileri için Ç1BD yöntemi kullanılarak yapılan HK işleminin başarımının literatürdeki diğer benzer HK yöntemleri ve geleneksel OMF ölçütü ile yapılan HK işlemi ile karşılaştırmalı sonuçları verilmiştir.

4.2.2 Ç1BD temelli HK donanımı mimarisi

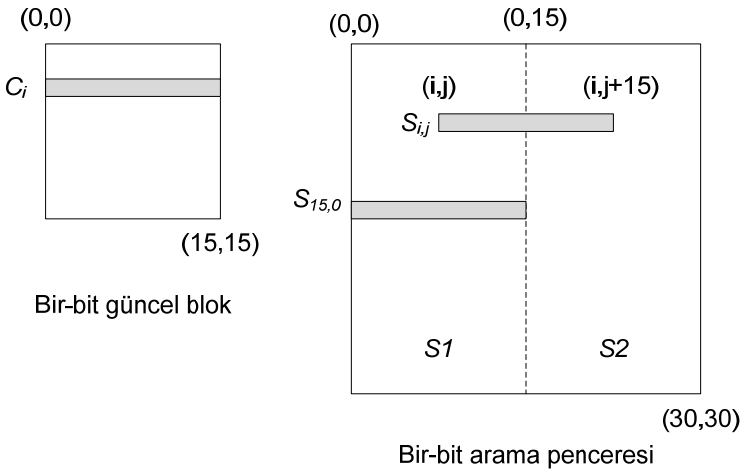
HVTDD ve KPTDD mimarilerinden 1BD temelli yöntemler için hangisinin kullanılmasının daha verimli olacağını belirlemek için her iki mimari de 1BD temelli bir HK yöntemi için gerçeklenmiştir. [48]'de, 1BD temelli HK için HVTDD mimarisi kullanılmıştır. Böylelikle, 8-bit/piksel gösterimi temelli HK mimarilerine göre son derece küçük yonga alanı kullanan bir mimari elde edilmiştir. KPTDD mimarisi geçmişte 1BD temelli bir HK yöntemine uygulanmamıştır. Bu tez çalışması kapsamında her iki mimarinin karşılaştırılması için KPTDD kullanılarak Ç1BD yönteminin gerçeklenmiş ve elde edilen sonuçların [48]'de önerilen mimari ile karşılaştırılmıştır. Karşılaştırma işlemi için [48]'deki mimari mevcut tasarım araçları kullanılarak gerçeklenmiştir.

4.2.2.1 Ç1BD temelli HK yönteminin HVTDD mimarisi ile gerçeklenmesi

Tablo 3.1 ve Tablo 3.2'de sırasıyla HVTDD ve KPTDD mimarilerinin 8-bit/piksel gösterimi temeli bir HK işleminde kullandıkları veri akışı yapısı gösterilmişti. Bu tablolardan anlaşılacağı üzere her İP, bir saat darbesinde bir aday ve bir güncel blok pikseli için işlem yapabilmektedir. Bir İP'nin, bir aday konum için MFT ölçütünü hesaplaması 256 saat çevrimi sürmektedir. Ancak, 1B sistolik dizide 16 adet İP bulunduğundan, her iki mimaride de 256 saat çevriminde 16 farklı aday konum için MFT ölçütü hesaplanabilmektedir. 16×16 blok boyutu ve $[-8,+7]$ arama aralığı için toplan 256 aday konum olduğundan bu mimarilerin bir tane güncel bloğa ait hareket

vektörünü hesaplayabilmeleri için $16 \times 16 \times 16 + 11 = 4111$ saat darbesi gerekmektedir.

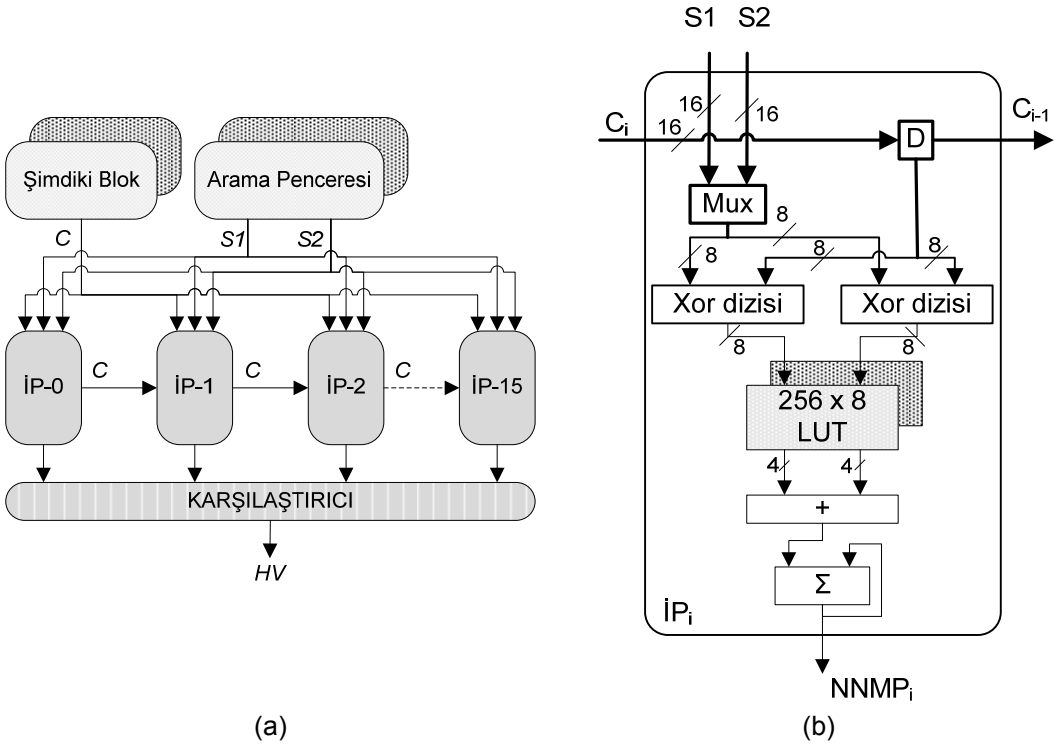
1BD temelli HK uyum ölçütü hesaplanırken BDUY kullanılmaktadır. Bu yöntemde güncel bloktaki ikili değerli piksel ile aday bloktaki ikili değerli pikselin XOR işlemine tabi tutulması yeterlidir. [48]'de önerilen mimaride, güncel blok ve aday bloktan pikseller, teker teker değil de 16 piksellik vektörler halinde okunmaktadır. Dolayısıyla 8-bit/piksel temelli yaklaşıma göre 16 katlık bir hız artışı sağlanmaktadır. Şekil 4.3'de bu yapı gösterilmiştir. Görüldüğü gibi güncel bloktan C_i şeklinde 16 piksellik bir satır vektörü okunurken arama penceresinden de $S_{i,j}$ şeklinde yine 16 bitlik bir satır vektörü okunmaktadır. Şekilde görülen vektörlerin büyük harfle yazılmasındaki amaç önceki bölümlerdeki piksel gösterimleri ile karıştırılmamasıdır.



Şekil 4.3: 16×16 blok boyutu ve $[-8,+7]$ arama aralığı için güncel blok ve arama penceresindeki piksel koordinatları

Ç1BD temelli HK yönteminin HVTDD mimarisi ile gerçeklenmesi ile oluşan yapı Şekil 4.4'de görülmektedir. Bu mimarinin [48]'de önerilen mimariden bir farkı yoktur. Çünkü Ç1BD ile [48] arasındaki fark, bit-düzlemi uyumlamasında değil süzgeçleme işlemindedir.

Şekil 4.4(a)'da görülen mimarinin geleneksel 8-bit/piksel temelli mimariden yapısal bir farkı yoktur. Ancak bellekten çıkan hattın genişliği 8 bit yerine 16 bittir çünkü artık bellekten 8-bitlik pikseller yerine 16 bitlik piksel vektörleri okunmaktadır. Şekil 4.4(b)'de ise Şekil 3.2(b)'den farklı olarak mutlak değer öbeği yerine 1BD yöntemindeki uyumlama işleminin gerçekleştirmek üzere basit XOR dizisi bulunmaktadır. Bu dizi 16 adet XOR kapısından oluşur ve yonga üzerinde çok küçük bir alan işgal eder.



Şekil 4.4: Ç1BD temelli HK yönteminin HVTDD mimarisine uygulanması a) Öbek gösterimi b) İP öbeğinin iç yapısı

Ç1BD yöntemi kullanılarak gerçeklenen HVTDD mimarisinde kullanılan veri akışı yapısı Tablo 4.2'de görülmektedir.

Tablo 4.2: 16×16 blok boyutu ve $[-8,+7]$ arama aralığı için Ç1BD temelli HK işlemi HVTDD temelli bir mimari ile gerçeklendiğinde kullanılan veri akışı yapısı

Çevrim	Giriş Verisi			İşlemci Girişleri		
Zamanı	*C*	*S1*	*S2*	**İP-0**	…	**İP-15**
0	*C0*	*S0,0*		*C0-S0,0*		
1	*C1*	*S1,0*		*C1-S1,0*		
2	*C2*	*S2,0*		*C2-S2,0*		
.		.		.		
.		.		.		
.		.		.		
14	*C14*	*S14,0*		*C14,S14,0*		
15	*C15*	*S15,0*		*C15-S15,0*	...	*C0-S15,0*
16	*C0*	*S16,0*	*S0,1*	*C0-S0,1*	...	*C1-S16,0*
17	*C1*	*S17,0*	*S1,1*	*C1-S1,1*	...	*C2-S17,0*
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
30	*C14*	*S30,0*	*S14,1*	*C14-S14,1*	...	*C15-S30,0*
31	*C15*		*S15,1*	*C15-S15,1*	...	*C0-S15,1*

Tabloda görülen *C0,* güncel bloğun birinci satırındaki 16 piksellik satır vektörünü, *S0,0* ise arama penceresinin birinci satır ve birinci sütunundan başlayan ve sağa doğru 16 bit uzunluğundaki satır vektörünü temsil etmektedir. Tablodan da görüldüğü gibi, bir aday bölgenin UNS ölçütünün hesaplanabilmesi için 16 saat darbesi yeterlidir. Toplamda ise bir MB'nin hareket vektörünün hesaplanabilmesi için $16\times16+15=271$ saat darbesine ihtiyaç vardır.

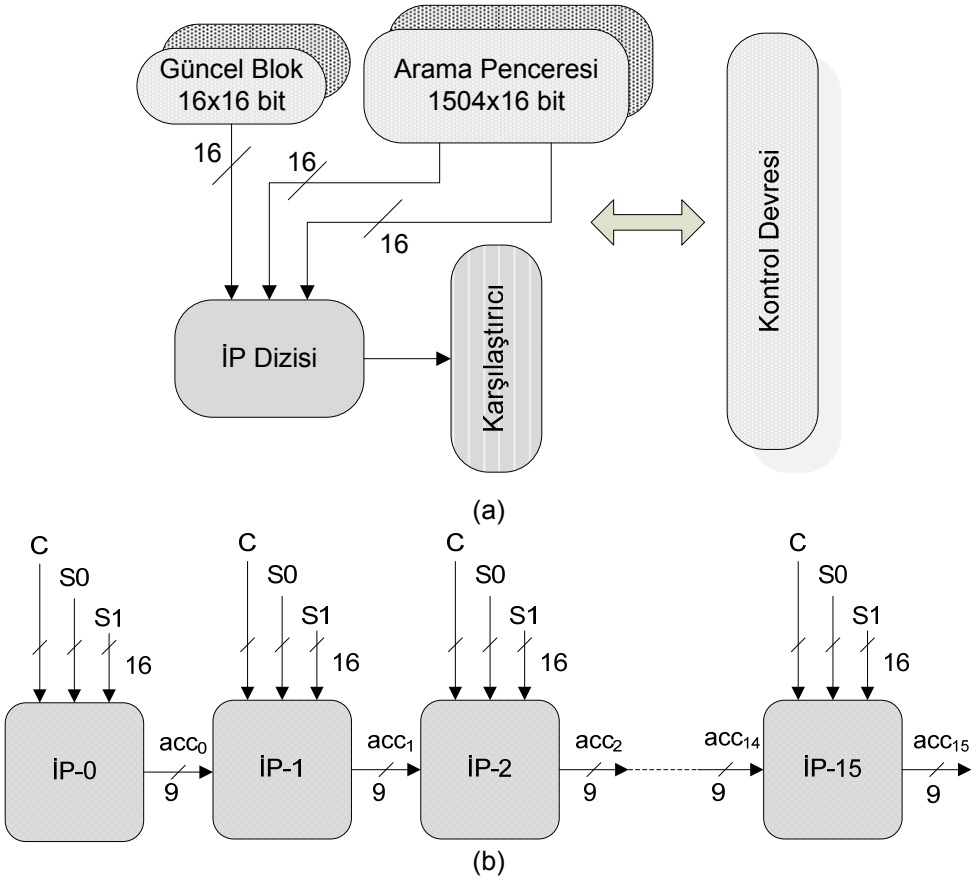
4.2.2.2 Ç1BD temelli HK yönteminin KPTDD mimarisi ile gerçeklenmesi

Bölüm 3.4.2'de açıklandığı üzere, KPTDD mimarisinde her bir İP, güncel bloğun belirli bir pikseli için mutlak fark ve kısmi uyum ölçütünü hesaplar. HK yöntemi, 1BD temelli bir yöntem olduğunda, bu durum değişiklik gösterir. 1BD temelli HK yöntemlerinde, piksel temelinde işlem yapmak yerine piksel vektörü temelinde işlem yapıldığından bu tip yöntemleri gerçekleyen mimarilerde bulunan bir İP öbeği, güncel bloktaki belirli bir satır için kısmi uyum ölçütünü hesaplar.

KPTDD mimarisinde HVTDD mimarisinden farklı olarak her bir İP'de, bir aday blok için uyum ölçütünü hesaplamak yerine ortaklaşa bir hesaplama işlemi söz konusudur. Dolayısıyla bu mimaride, karşılaştırıcıya bütün İP öbeklerinden çıkış gelmesi söz konusu değildir. Sadece sistolik dizideki en son İP öbeği karşılaştırıcıya girecek veriyi, yani uyum ölçütünü hesaplar.

Uyum ölçütü ardışık İP'ler arasından toplanarak hesaplanır. Bir aday konumun uyum ölçütü, 16 saat darbesinde hesaplanır. Paralel çalışma sayesinde her saat darbesinde, bir aday konumun uyum ölçütü hesaplanabildiği için HVTDD mimari ile KPTDD mimarisinin işlem kapasitesi birbirine eşittir.

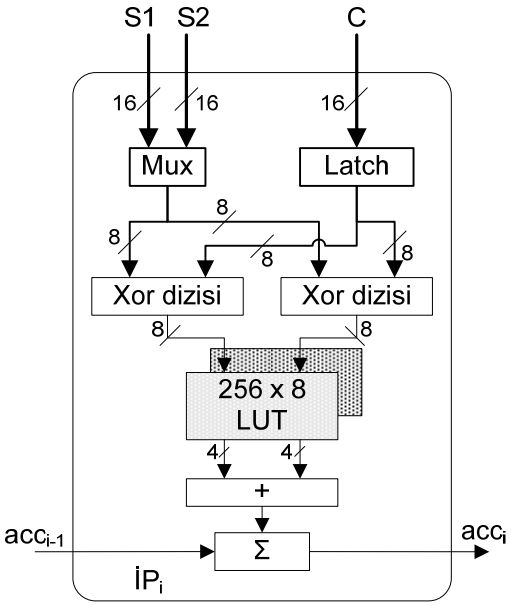
Ç1BD temelli HK yönteminin bu tez kapsamında KPTDD mimarisi ile gerçeklenmesi sonucu elde edilen yeni HK donanımı ve sistolik dizi yapısı Şekil 4.5'de görülmektedir. Burada görülen yapıda, güncel blok verisi ve arama penceresi verisi İP'lerin tümüne birden girmektedir. Bu nedenle bu mimari yarı sistolik olarak da nitelendirilebilir. Sistolik dizilerde ilk İP'ye giren verilerin takip eden saat darbelerinde seri bir biçimde diğer İP öbeklerine de gönderildiği düşünüldüğünde yarı sistolik nitelemesi çok da yanlış olmaz.



Şekil 4.5: a) Ç1BD temelli HK yöntemi için önerilen KPTDD mimarisi b) İP dizisi öbeği

Mimari daha düşük seviyede incelenirse, her bir İP öbeğinin iç yapısı Şekil 4.6'da görülmektedir. HVTDD mimarisinde olduğu gibi buradaki İP yapısında da yine bir XOR dizisi, bir adet okuma tablosu ve okuma tablosu çıkışında bir toplayıcı vardır.

Şekil 3.3(b)'den farklı olarak burada kısmi uyum ölçütü bir önceki İP öbeğinden gelen kısmi uyum ölçütü ile toplanmaktadır. HVTDD mimarisindeki İP öbeğinde, bu işlem paylaşımlı değil de her İP bir UNS ölçütü hesaplayacak şekilde gerçekleştirilmektedir. Bu durumun sağlayacağı avantajların açıklanabilmesi için KPTDD mimarisinde kullanılan veri akışı yapısına bakmak gerekmektedir.



Şekil 4.6: KPTDD mimarisinin Ç1BD temelli HK yöntemine uygulanması için tasarlanan İP mimarisi

Tablo 4.3'de, KPTDD mimarisinde kullanılan veri akışı yapısı görülmektedir. İlk olarak, bütün İP öbeklerinin çalışır hale gelebilmesi için 15 saat darbesinin geçmesi gerekmektedir. Bu süre zarfında sırasıyla birinci İP'den başlayarak, takip eden saat darbelerinde İP'ler içerisindeki *Latch* öbeklerine güncel blok verisi satır satır yerleştirilir. Güncel blok verisinin *Latch* öbeklerine yerleştirilmesi esnasında, İP öbekleri uyum ölçütü için gerekli hesaplamaları yapmayı sürdürür. HK için yapılan işlemler sırayla aşağıdaki şekilde gerçekleşmektedir:

- Öncelikle arama penceresinden gelen 16 bitlik satır vektörleri, kontrol sinyallerinin denetimindeki çoğullayıcı aracılığıyla seçilir.

- Ardından güncel blok verisi ile çoğullayıcı çıkışındaki 16 bitlik vektörler XOR dizisinin girişine uygulanarak uyumlama işlemi gerçekleştirilir.

- Uyumlama işleminin sonucunda XOR dizisi çıkışındaki 16 bitlik vektör,

içerisindeki birlerin sayılması için LUT öbeğinin girişine uygulanır. LUT, girişindeki bu 16 bitlik sayıyı 8 bitlik iki parça halinde işler. 16 bitlik bir sayının alabileceği olası değerlerler ile 8 bitlik bir sayının alabileceği olası değerler arasında 256 kat fark vardır. Sonuç olarak 256×256 girişli tek bir LUT yerine 256 girişli iki LUT kullanılarak yonga alanından tasarruf edilir.

- Daha sonra LUT çıkışındaki iki adet 4 bitlik sayı, bir toplayıcı yardımıyla toplanır. Böylece İP öbeği, girişlerindeki verilere ait kısmi UNS'yi hesaplamış olur. Bu işlemlerin yapıldığı saat darbesinde bir önceki İP öbeğinin hesaplamış olduğu kısmi UNS değeri, güncel İP'nin kısmi UNS girişine yazılmıştır. Bu değer bir toplayıcı aracılığıyla, yeni elde edilen kısmi UNS değeri ile toplanarak bir sonraki İP öbeğinin ilgili girişine yazılır.

- Kısmi UNS değeri bu şekilde 16. İP öbeğinin kısmi UNS girişine kadar toplanarak gittiğinden, bu öbeğin çıkışında bir aday konum için elde edilecek olan UNS değeri bulunur. Bu değer karşılaştırıcı girişinden alınır ve denetim sinyalleri doğrultusunda karşılaştırma işlemi yapılır. Tüm aday konumlar için bu işlem tamamlandığında elde edilen sonuç, denetim birimine gönderilerek hareket vektörü bilgisi üretilir.

Tablo 3.1'de olduğu gibi Tablo 4.3'de de köşeli parantezler içerisine alınmış veriler görülmektedir. Bu gösterim, ilgili verinin artık YÜ bellekten değil de ilgili İP içerisindeki *Latch* öbeğinden okunduğunu göstermektedir. Bu durumda, YÜ bellek bant genişliğinde büyük miktarlarda bir düşüş gözlenmektedir. Geleneksel 8-bit/piksel gösterimi temelli yöntemlerde de YÜ bellek bant genişliği HVTDD mimarisine göre düşürülebilmektedir. Ancak 8-bit/piksel gösterimi temelli yöntemlerde 1B sistolik dizi ile bu tasarrufu yapmak mümkün değildir ve 2B bir sistolik dizi kullanılması gerekir. Bu nedenle 1BD temelli HK yöntemleri, mimari tasarımı kolaylaştırmakta ve donanımsal gereksinimlerin daha rahat karşılanmasını sağlamaktadır.

Tablo 4.3: 16×16 blok boyutu ve $[-8,+7]$ arama aralığı için Ç1BD temelli HK işlemi KPTDD temelli bir mimari ile gerçeklendiğinde kullanılan veri akışı yapısı

Çevrim	Giriş Verisi			İşlemci Girişleri					
Zamanı	*C*	*S1*	*S2*	İP-0	İP-1	. . .	İP-14	İP-15	Hesaplanan Blok Bozlumasının Ait Olduğu Konum
0	C0	S0,0		C0-S0,0					
1	C1	S1,0		[C0]-S1,0	C1-S1,0				
2	C2	S2,0		[C0]-S2,0	[C1]-S2,0				
.	.	.		.	.				
.	.	.		.	.				
.	.	.		.	.	.			
14	C14	S14,0		[C0]-S14,0	[C1]-S14,0	…	C14-S14,0		
15***	C15	S15,0		[C0]-S15,0	[C1]-S15,0	…	[C14]-S15,0	C15-S15,0	D(0,0)
16	C0	S16,0		[C0]-S16,0	[C1]-S16,0	…	[C14]-S16,0	[C15]-S16,0	D(1,0)
.	.			.	.	.	.	.	.
.	.	.		.	.	.	.	.	.
.	.	.		.	.	.	.	.	.
31	C15	S31,0		[C0]-S31,0	[C1]-S31,0	…	[C14]-S31,0	[C15]-S31-0	D(16,0)
32	C0	S32,0	S0,1	[C0]-S0,1	[C1]-S32,0	…	[C14],S32,0	[C15]-S32,0	D(17,0)
.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.
46	C14	S46,0	S14,1	[C0]-S14,0	[C1]-S14,0	…	[C14]-S14,1	[C15]-S46,0	D(31,0)
47	C15		S15,1	[C0]-S15,1	[C1]-S15,1	…	[C14]-S15,1	[C15]-S15,1	D(0,1)
***Bu andan itibaren bütün İP' ler çalışmaktadır									

[48]'de önerilen mimaride, tüm çalışma süresi boyunca güncel blok belleğinden veri okunması gerekmektedir. Bellek yazma okuma işlemlerinin güç tüketimini artırıcı en önemli etkenlerden birisi olduğu düşünülürse bu oranın düşürülmesi şarttır. Bu tez çalışması kapsamında önerilen mimaride, devrenin çalışmaya başlamasını takip eden ilk 16 saat darbesinin ardından güncel blok belleğinden okuma işlemi yapılmasına gerek kalmamaktadır. 16×16 blok boyutu ve $[-16,+15]$ arama aralığı için toplam 1024 farklı aday konum vardır. [48]'de önerilen mimari ile HK işlemi toplam 1039 saat çevriminde tamamlanabilmektedir ve tüm bu zaman boyunca güncel blok belleğinden okuma yapılması gerekmektedir. Önerilen mimaride ise sadece ilk 16 saat darbesinde bu bellekten okuma yapılması yeterlidir. Kalan 1023 saat darbesi boyunca güncel blok verileri İP öbeklerindeki *Latch* devresinden okunmakta, dolayısıyla YÜ bellekten veri okunmasının azaltılması ile güç tüketimi tasarrufu sağlanması mümkündür.

4.2.2.3 Ç1BD temelli HK yönteminin HVTDD ve KPTDD mimarileri kullanılarak gerçeklenmesi ile elde edilen sonuçlar

Önerilen KPTDD mimarisinin gerçeklenmesi ile elde edilen sonuçların diğer mimarilerle karşılaştırılabilmesi için bu tez çalışması kapsamında [48]'de önerilen mimari, güncel araçlar kullanılarak yeniden gerçeklenmiştir. Gerçekleme işleminde Verilog donanım tanımlama dili (HDL) kullanılmıştır. Bu işlemin ardından davranış benzetimleri yapılmış ve kavramsal çalışabilirlik gösterilmiştir.

Kavramsal benzetim ardından tasarım XC2VP30 FPGA yongası üzerinde çalıştırılmıştır. Bu aşamada öncelikle yazılan kodlar hedef FPGA yongası için sentezlenmiştir. Sentezleme işleminde Synplify Premier [112] aracı kullanılmış ve hedef olarak da XC2VP30 FPGA yongası seçilmiştir. Tablo 4.4'de, [48]'de önerilen HVTDD mimarisi ile 1BD temelli yöntemler için ilk kez bu çalışmada önerilen KPTDD mimarisinin sentezleme raporlarının özetleri sunulmuştur.

Tablo 4.4: Sentez sonuçları

HK Yöntemi	Ç1BD	
Mimari	Önerilen	[48]
Frekans (MHz)	196	98
Kullanılan LUT sayısı Yonga yararlanımı	1121 (3%)	2541 (8%)
Kullanılan Blok RAM sayısı	16	4
Kullanılan İki uçlu RAM sayısı	16	16
Kullanılan ROM sayısı	24	96

Tablo 4.4'de görülen sentez sonuçları incelendiğinde, önerilen yöntemin çalışabildiği maksimum saat frekansı ile [48]'de önerilen mimarinin çalışabildiği en yüksek saat frekansı arasında neredeyse iki kat fark vardır. Sonuç olarak, önerilen mimarinin güncel sentez araçları ile sentezlenmesinin daha kolay olduğu dolayısıyla sahip olduğu donanım karmaşasın daha az olduğu ortaya çıkmaktadır. Sentez sonuçlarına göre İki mimarideki karşılaştırıcı öbeklerinin arasında neredeyse iki katlık bir donanım alanı farkı olduğu gözlenmiştir. KPTDD mimarisindeki karşılaştırıcı 94 LUT'lik bir alan kaplarken, HVTDD mimarisindeki karşılaştırıcı 160 LUT'lik bir alan kaplamıştır.

Önerilen mimarinin güç tüketiminin HVTDD mimarisine göre daha düşük olduğunu göstermek amacıyla FPGA üretici firmalarının sağladığı yazılımlardan faydalanılmıştır. Bu çalışma kapsamında, Xilinx marka bir FPGA kullanıldığı için yine

bu firmanın sağlamış olduğu XPower güç tüketimi kestirimi aracı kullanılarak bir dizi güç tüketimi analizi gerçekleştirilmiştir. Bu aracı kullanmak için devrenin çalışması esnasında, içerisindeki kapıların anahtarlama etkinliklerinin bilinmesi gerekmektedir. Bu etkinlikleri gözlemleyebilmek için Mentor Graphics'in ModelSim benzetim aracı [113]'de gösterilen şekilde kullanılmıştır. Bu çerçevede, nihai yerleştirme ve yönlendirmeden (post place&route) sonra elde edilen devre listesinin (netlist) benzetimleri yapılarak gerekli anahtarlama aktivitelerini içeren dosyalar üretilmiştir. Elde edilen dosya, güç tüketimi kestirimi aracına sokulmuştur. Güç tüketimi analizinin en kötü koşullardaki çalışmayı yansıtabilmesini sağlamak amacıyla hareket vektörü konumlarının merkezden oldukça uzak olduğu güncel blok konumları seçilmiştir. Tablo 4.5'de örnek analiz sonuçları, bazı hareket vektörleri ve ilgili güncel blok konumları görülmektedir. Test için "*footbal*l" dizisinin 75. ve 100. çerçeveleri aday çerçeve olarak seçilerek tabloda belirtilen bloklar için HK işlemi yapılmıştır.

Tablo 4.5: Ç1BD temelli HK yöntemi için elde edilen güç tüketimi sonuçları

Blok Konumu, Çerçeve Numarası	Hareket Vektörü	[48] (mW) HVTDD (66 MHz)	Önerilen (mW) KPTDD (66 MHz)	Önerilen (mW) KPTDD (166 MHz)
(15,11), 100	(2,-2)	332	168	235
(12,10), 75	(4,-2)	287	155	201
(12,12), 75	(6,6)	209	129	187
(12,8), 75	(-1,-2)	240	145	189
(20,8), 75	(-13,2)	418	199	300
(18,8), 100	(-1,-3)	313	166	227
Enerji		4543 mW.ns	2429 mW.ns	1343mW.ns

Tablo 4.5'de her iki mimarinin, 66MHz elde edilen güç tüketimi değerleri görülmektedir. Bu tez çalışmasında önerilen, KPTDD temelli donanım mimarisinin 166 MHz çalışma frekansı için elde edilen güç tüketimi değerleri de görülmektedir. 66 MHz için elde edilen değerlere bakıldığında bu tez çalışmasında önerilen mimarinin [48]'de önerilen donanım mimarisinden aynı şartlarda ortalama 1.87 kat daha az güç tüketimine sahip olduğu görülecektir. Yine bu tez çalışmasında önerilen donanım mimarisi, 166MHz'de çalıştırıldığında [48]'de önerilen mimarinin 66MHz çalışma frekansında sahip olduğu güç tüketiminden daha düşük miktarda güç harcamıştır. Tablo 4.5'de ayrıca mimarilerin kullandıkları enerji de görülmektedir. [48]'de önerilen mimari en yüksek enerjiyi harcarken bu tez çalışmasında önerilen donanım mimarisinin harcadığı enerji daha azdır ve çalışma frekansı arttığında ihtiyaç duyulan enerji daha da azalmaktadır.

Tablo 4.6'da, bu tez çalışmasında önerilen Ç1BD temelli HK donanımı mimarisi ile literatürdeki güncel çalışmalardan [99]'da önerilen 8bit/piksel gösterimi temelli HK donanımı mimarisi karşılaştırılmıştır. Bu tez çalışmasında, kullanılan HK yönteminin tamamen ikili olması nedeniyle, [99]'da önerilen 8bit/piksel gösterimi temelli HK mimarisine göre çok az sayıda İP kullanılmaktadır. Bunun yanında [99]'da önerilen mimaride MFT ölçütünün hesaplanabilmesi için 2B paralel toplayıcı ağacı kullanılmaktadır. Bu tez çalışmasında önerilen donanım mimarisinde ise kısmi UNS değerleri son derece basit bir şekilde, ardışık İP'ler arasında toplanarak 16. İP çıkışında nihai UNS değeri elde edilmektedir.

Tablo 4.6: [99]'da önerilen 8bit/piksel gösterimi temelli HK donanımı mimarisi ile bu tez çalışmasında önerilen Ç1BD temelli HK donanımı mimarisinin karşılaştırılması.

	[99]'da önerilen mimari	Bu tez çalışmasında önerilen mimari (KPTDD)
Toplam İP Sayısı	256	16
Arama Aralığı	[-16,16]	[-16,15]
Blok Boyutu	Değişken	16×16
Saat Darbesi Sayısı	1129	1039
Kullanılan Teknoloji	0.18µm	FPGA
Maksimum Frekans	200MHz	196MHz
Güç Tüketimi	$423mW@180MHz$	$223mW@166MHz$
Performans	$720p@45çerçeve/saniye$	$720p@45çerçeve/saniye$

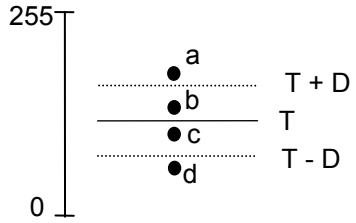
Önerilen mimarinin gerçek zamanlı çalışma performansı için sentez sonuçlarına bakılarak yaklaşık bir kestirim yapılabilir. Buna göre XC2VP30 FPGA yongası için önerilen mimarinin çalışma frekansında, 16×16 blok boyutu, $[-16,+15]$ arama aralığı, $720p$ HDTV çözünürlüğünde ve $45\ çerçeve/saniye$ hızında HK işlemi yapmak mümkün olmaktadır. Bu hesap şu şekilde yapılmaktadır; (720*1280) [çerçeve boyutu]/(16*16) [blok boyutu]*1039 [saat darbesi]*45 [çerçeve/saniye]=168MHz. Önerilen yöntemin benzetimi kolaylık olması açısından 166MHz de gerçekleştirilmiştir.

4.3 K-1BD Temelli HK Yöntemi ve Donanım Mimarisi

1BD dönüşümü yönteminde, güncel blok ile aday blok arasında karşılıklı piksellerin uyumsuzluğuna bakılırken K-1BD yönteminde, daha doğru bir değerlendirme yapılmasını sağlayabilmek için bir kısıt maskesi kullanılması önerilmiştir. [50]'de önerilen 2BD kullanan yönteme göre daha yüksek başarım ve daha az işlem yükü elde ettiği gösterilmiştir [52].

4.3.1 K-1BD Temelli HK yöntemi

1BD kullanan HK yöntemlerinin en büyük sorunu birbirlerine çok yakın ancak eşiğin zıt taraflarında bulunan piksellerin farklı kabul edilmesidir. [52]'de bu sorunu ortadan kaldıran kısıtlanmış 1-Bit dönüşümü yapısı önerilmiştir. [52]'de, 1-Bit dönüşümü yapılırken eşik değerine ek olarak bir de kısıt maskesi kullanılmaktadır. Bu kısıt maskesi ile piksellerin, dönüşüm eşiğine göre kabul edilebilir bir mesafede olup olmadıklarına bakıldıktan sonra uyumlama işleminde hesaba katılıp katılmayacaklarına karar verilir. Şekil 4.7'de sadece eşikleme işlemi kullanılarak piksellerin farklı sınıflara ayrılmasının yaratabileceği olası sorun gösterilmeye çalışılmıştır.



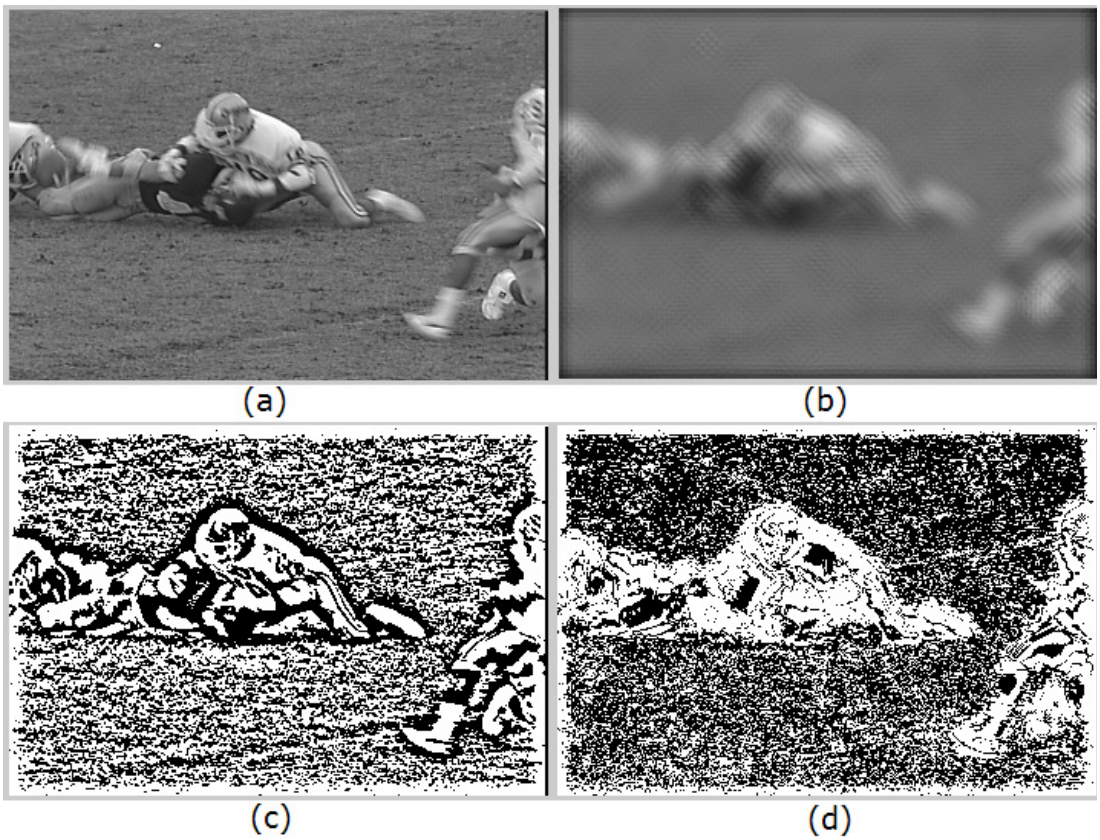
Şekil 4.7: Kısıtlanmış 1-Bit dönüşümü yönteminin temel çalışma mantığı

K-1BD yönteminde, 1BD işlemi geleneksel 1BD işlemi ile aynıdır. Bu yöntemin asıl farklılığı UNS ölçütünü hesaplarken kullanılan yaklaşımdır. Yeni uyumlama ölçütü için fazladan bir kısıt maskesine ihtiyaç vardır. Bu maskenin nasıl oluşturulduğunu anlamak için Şekil 4.7'ye bakılacak olunursa, *b* ve *c* pikselleri birbirlerine çok yakın genlik değerinde olsalar da XOR işleminin sonucu *1* üretecektir. Dolayısıyla bu iki nokta 1BD temelli HK yönteminde uyumsuz olarak hesaba katılacaktır. Bu tip pikseller için bu yanlış bir karardır. K-1BD yöntemi bu durumun giderilmesi amacıyla önerilmiştir. K-1BD'de kısıt maskesi *CM* (4.3)'de görüldüğü gibi hesaplanmaktadır.

$$CM(i,j) = \begin{cases} 1, & eğer \ \left| I(i,j) - I_F(i,j) \right| \geq D \\ 0, & diğer \end{cases} \tag{4.3}$$

(4.3)'den anlaşılacağı gibi CM imge boyutunda bir matristir ve elemanları 1 ve 0'lardan oluşmaktadır. Bu değerler asıl video çerçevesindeki pikseller ile süzgeçlenmiş video çerçevesindeki pikseller arasındaki mutlak farkın önceden belirlenmiş bir eşikten büyük olup olmadığına göre belirlenir. Bütün bu işlemler örnek bir video çerçevesi için Şekil 4.8'de *"football"* dizisi kullanılarak gösterilmiştir.

Kısıt maskesi elde edildikten sonra geriye kalan iş, UNS ölçütünün kısıt bilgisini değerlendirebilecek şekilde yeniden düzenlenmesidir. Eşitlik (4.4)'de kısıtlanmış UNS (KUNS) (CNNMP – Constrained Number of Non-Matching Points) olarak adlandırılan bu yeni uyumlama ölçütü görülmektedir.



Şekil 4.8: a) *"football"* dizinden bir örnek çerçeve, b) Eşitlik 4.1'deki çekirdek ile süzgeçlenmiş imge, c) İmge çerçevesinin 1BD uygulanmış hali, d) Çerçeveye ait kısıt maskesi

Eşitlik (4.4)'deki "$\bullet$" sembolü mantıksal VE işlemini ve "||" sembolü de mantıksal VEYA işlemini temsil etmektedir, geriye kalan işlemler eşitlik (4.2) ile aynıdır.

$$KUNS(m,n)=\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left\{\begin{array}{l}\left[CM^{t}(i,j)\,||\,CM^{t-1}(i+m,j+n)\right]\\ \bullet\left[B^{t}(i,j)\oplus B^{t-1}(i+m,j+n)\right]\end{array}\right\} \qquad (4.4)$$

$$-s\le m,n\le s-1$$

Bu eşitlik kullanılarak elde edilen en küçük değer, ilgili güncel bloğa ait hareket vektörü olarak belirlenir. Eğer burada Şekil 4.7'de görülen *b* ve *c* piksellerinin durumu incelenecek olursa; bu pikseller için kısıt maskesinden 0 geleceği açıkça görülecektir. Sonuç olarak bu iki piksellerin KUNS ölçütü üzerindeki etkisi, yöntemin bu iki noktayı uyumlu kabul etmesi şeklinde olacaktır. Dolayısıyla bu iki piksel için XOR işleminin sonucunun uyumsuz nokta sayısına bir etkisi yoktur. Tablo 4.7, K-1BD yönteminin farklı HK yöntemleri ile karşılaştırmalı başarım sonuçları verilmiştir.

Tablo 4.7: Geçmişte önerilmiş düşük bit derinliğinde piksel gösterimine dayanan HK yöntemleri ile tam arama yapılarak yeniden oluşturulan dizilerin ortalama PSNR değerleri.

Yöntem	Video Dizileri (Dizi Adı, Çerçeve Boyutu, Dizi Uzunluğu)					
	"football" 352×240 125	"foreman" 352×288 300	"tennis" 352×240 150	Garden 352×240 115	"mobile" 352×240 300	"coastguard" 352×288 300
MFT	22.88	32.09	29.45	23.79	23.94	30.48
2BT [50]	22.06	30.70	28.46	23.43	23.66	29.94
1BD [48]	21.83	30.32	28.11	23.31	23.61	29.83
Ç1BD [51]	21.81	30.38	28.18	23.26	23.63	29.88
K-1BD[52]	22.10	30.86	28.71	23.38	23.69	29.98

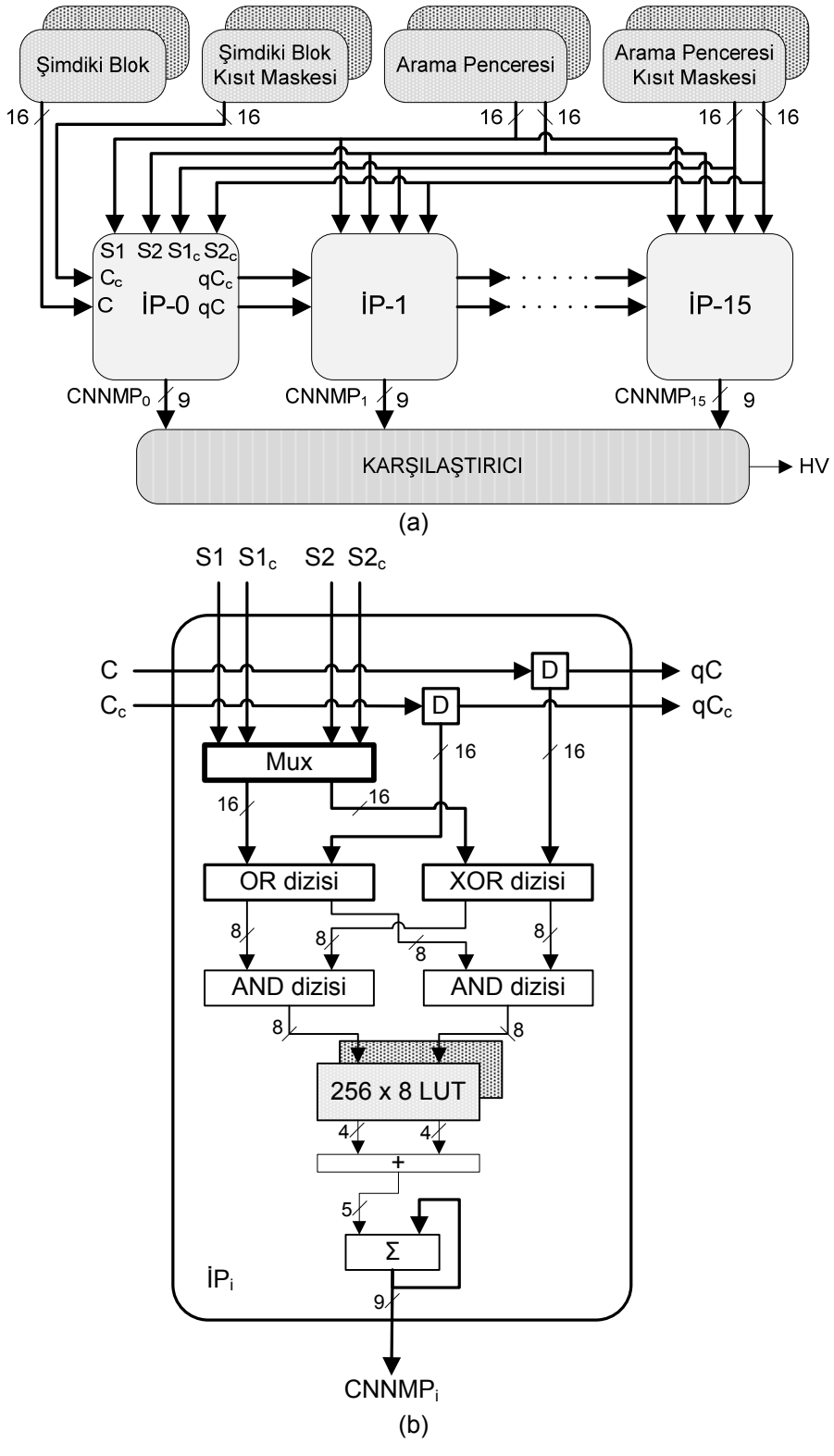
Tablodan da görüldüğü üzere, düşük bit derinliğine sahip piksel gösterimine dayanan güncel HK yöntemleri arasında K-1BD yönteminin başarımı en üst seviyededir. Tez çalışmasında bu nedenle K-1BD temelli HK yöntemi için de donanım mimarileri geliştirilmiştir.

4.3.2 K-1BD Temelli HK Donanımı Mimarisi

Bu bölümde K-1BD temelli HK yönteminin HVTDD ve KPTDD mimarileri kullanılarak gerçeklenmesi ve bu gerçekleme sonucu elde edilen sonuçlar ele alınmaktadır.

4.3.2.1 K-1BD temelli HK yönteminin HVTDD mimarisi ile gerçeklenmesi

K-1BD temelli HK yöntemi için HVTDD mimari kullanılarak tasarlanan donanım, Ç1BD yöntemi için tasarlanan donanımdan yapısal olarak iki temel farklılık içerir. Bunlardan ilki güncel blok ve arama penceresi için hesaplanan kısıt maskesi bellekleridir. Bir diğer farklılık ise İP mimarisinin K-1BD yöntemindeki uyumlama ölçütünü gerçekleştirebilecek şekilde değiştirilmiş olmasıdır. Şekil 4.9'da K-1BD yöntemi için tasarlanan mimari ve bu donanımın içerisinde kullanılan İP öbeğinin mimarisi görülmektedir. Geleneksel 1BD dönüşümü temelli mimarilerde kullanılan İP öbeklerinden farklı olarak, K-1BD yöntemindeki KUNS ölçütünün hesaplanması için tasarlanan İP öbeğinde XOR dizisine ek olarak iki adet 16 bitlik AND dizisi ve bir de 16 bitlik OR dizisi bulunmaktadır. Bunların dışında kısıtlanmış verilerin girişleri ve çıkışları için fazladan birer giriş ve çıkış ucu, ayrıca güncel blok verisinin sistolik dizideki diğer İP'lere gönderilmesi için fazladan bir adet *D* tipi yaz-boz kullanılmıştır.



Şekil 4.9: a) K-1BD temelli HK donanımının HVTDD mimarisi ile gerçeklenmesi, b) K-1BD donanımında kullanılan İP mimarisi

Şekil 4.9(b)'de İP öbeği içerisinde görülen çoğullayıcı yapısı geleneksel İP öbeklerindeki çoğullayıcı yapıları gibi iki giriş ve tek çıkışlı değildir. Aday blok için hesaplanan kısıt maskesinden gelen verilerin de düzenlenmesi gerektiği için burada

4 girişli ve 2 çıkışlı bir çoğullayıcı kullanılmıştır. Çoğullayıcı öbeği, tek bir öbekmiş gibi görünse de iki adet iki girişli tek çıkışlı çoğullayıcıdan başka bir şey değildir.

K-1BD temelli donanım mimarisinin çalışması ile Ç1BD temelli mimarinin çalışması arasındaki tek fark pikseller arasındaki uyumun hesaplanması aşamasındadır. Bu işlem dışındaki tüm adımlar birbirinin aynıdır.

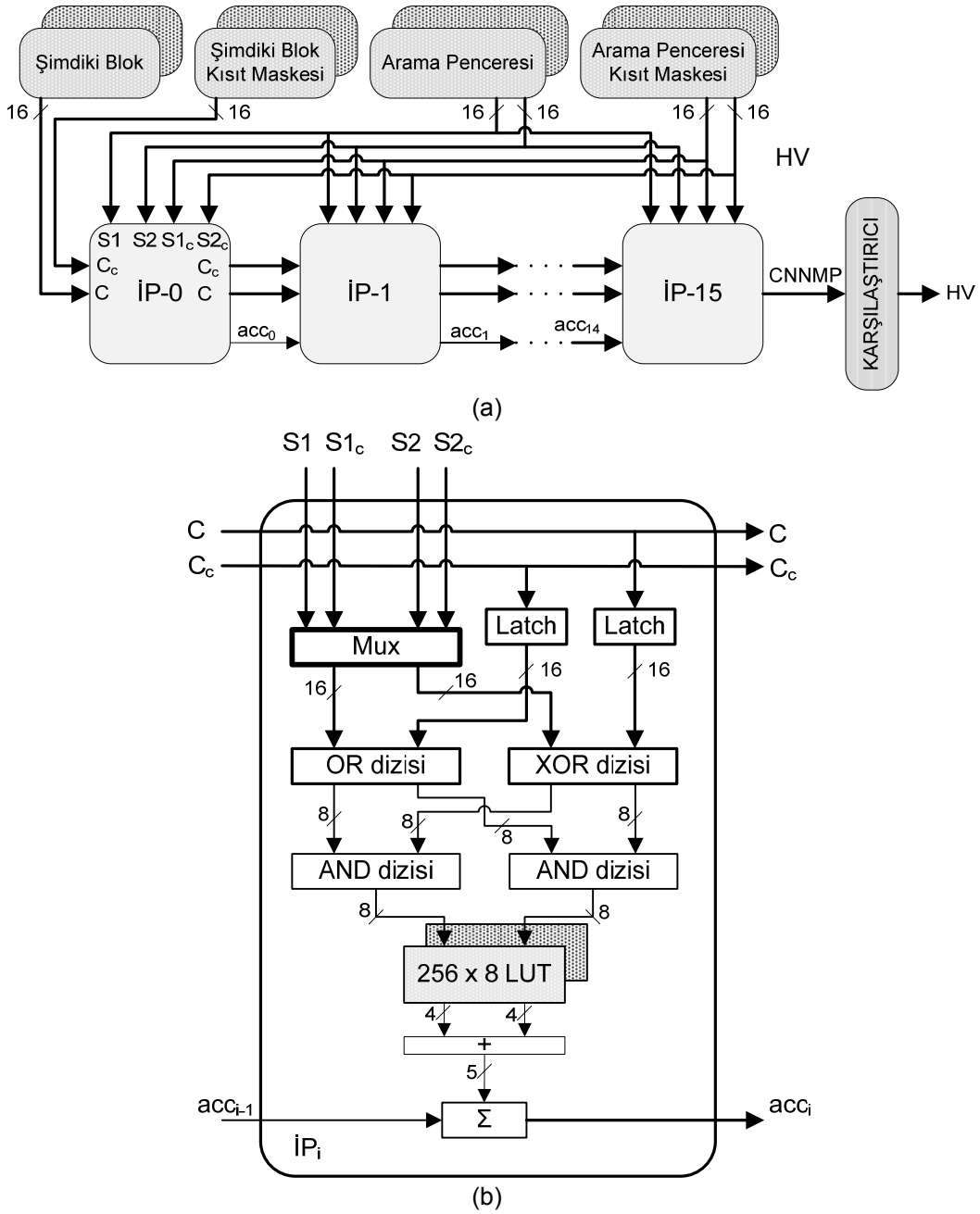
Tablo 4.8: HVTDD kullanılarak tasarlanan K-1BD temelli HK donanımında kullanılan veri akışı yapısı

Çevrim	Giriş Verisi						İşlemci Girişleri		
Zamanı	*C*	*CC*	*S1*	*CS1*	*S2*	*CS2*	İP-0	…	İP-15
0	C0	CC0	S0,0	CS0,0			C0-S0,0		
1	C1	CC1	S1,0	CS1,0			C1-S1,0		
2	C2	CC2	S2,0	CS2,0			C2-S2,0		
.			.	.			.		
.			.	.			.		
.			.	.			.		
14	C14	CC14	S14,0	CS14,0			C14,S14,0		
15	C15	CC15	S15,0	CS15,0			C15-S15,0	…	C0-S15,0
16	C0	CC0	S16,0	CS16,0	CS0,1	CS0,1	C0-S0,1	…	C1-S16,0
17	C1	CC1	S17,0	CS17,0	CS1,1	CS1,1	C1-S1,1	…	C2-S17,0
.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.
30	C14	CC14	S30,0	CS30,0	CS14,1	CS14,1	C14-S14,1	…	C15-S30,0
31	C15	CC15			CS15,1	CS15,1	C15-S15,1	…	C0-S15,1

Tablo 4.8'de, K-1BD yöntemi için HVTDD mimarisinde kullanılan veri akışı yapısı görülmektedir. Buradaki veri akışı yapısının Ç1BD yönteminde aynı mimari için kullanılan veri akışı yapısından farkı yoktur, sadece kısıt maskesinden gelen veriler de veri akışına eklenmiştir.

4.3.2.2 K-1BD temelli HK yönteminin KPTDD mimarisi ile gerçeklenmesi

K-1BD temelli HK yöntemini gerçekleştirecek donanımın KPTDD mimarisi kullanılarak tasarlanması, Ç1BD yöntemi için aynı mimari kullanılarak yapılan tasarıma göre küçük farklılıklar içerir. Bu farklılıklardan birisi, kısıt maskeleri için eklenecek fazladan bellek öbekleridir. Bir diğer farklılık da İP öbeklerinin ek giriş çıkış bağlantılarının yanı sıra KUNS ölçütünü gerçekleştirecek ek mantık kapılarının eklenmesidir. KPTDD mimarisi kullanılarak tasarlanan K-1BD temelli HK donanımının öbek gösterimi ve tasarlanan İP öbeğinin içyapısı sırasıyla Şekil 4.10(a) ve Şekil 4.10(b)'de görülmektedir.



Şekil 4.10: a) KPTDD mimarisi kullanılarak tasarlanmış K-1BD temelli HK donanımı mimarisi, b) Kullanılan İP mimarisi

Burada kullanılan İP öbeğinde HVTDD mimarisindekinden farklı olarak KUNS değerinin hesaplanmasındaki farklılığı gerçekleştirebilmek için LUT çıkışının kısmi uyum ölçütü ile toplanması işlemi yapılmaktadır. Daha önce açıklandığı gibi, saat darbesi ile birlikte toplanan LUT çıkışları, bir önceki İP öbeğinden gelen kısmi uyum ölçütü ile toplanarak bir sonraki saat darbesinde kullanılmak üzere bir sonraki İP öbeğine gönderilmektedir.

4.3.2.3 K-1BD temelli HK yönteminin HVTDD ve KPTDD mimarileri kullanılarak gerçeklenmesi ile elde edilen sonuçlar

Bu bölümde K-1BD temelli HK yöntemi için önerilen KPTDD mimarisinin gerçeklenmesi ile elde edilen sonuçların [48]'de önerilen mimari ile karşılaştırılabilmesi için Ç1BD yönteminde izlenen adımlar tekrarlanmıştır. Bu nedenle, bu bölümde sadece K-1BD yöntemi için elde edilen gerçekleme sonuçları sunulacaktır. Karşılaştırma amacıyla tablolarda Ç1BD yöntemi için elde edilen sonuçlara da yer verilmiştir.

Tablo 4.9'da, K-1BD temelli HK yöntemi için elde edilen sonuçlar Ç1BD temelli HK yöntemi için elde edilen sonuçlarla birlikte sunulmuştur. Sentez raporu incelendiğinde K-1BD için KPTDD mimarisinin çalışma frekansının bir miktar düştüğü görülmektedir. Bunun temel sebebi, KUNS ölçütünün gerçekleştirilebilmesi için İP öbeğine eklenen fazladan OR ve AND dizileridir.

Tablo 4.9: Sentez sonuçları

HK Yöntemi	Ç1BD		K-1BD	
Mimari	Önerilen KPTDD	[48] HVTDD	Önerilen KPTDD	[48] HVTDD
Frekans (MHz)	187	98	179	92
Kullanılan LUT sayısı Yonga yararlanımı	1121 (3%)	2541 (8%)	1721 (5%)	3064 (10%)
Kullanılan Blok RAM sayısı	16	4	20	8
Kullanılan İki uçlu RAM sayısı	16	16	32	32
Kullanılan ROM sayısı	24	96	27	96

Tablo 4.9'a bakıldığında Ç1BD ve K-1BD donanımı mimarilerinin benzer en yüksek çalışma frekansına sahip oldukları görülmektedir. K-1BD donanımı mimarisi de, XC2VP30 FPGA yongası için, 16×16 blok boyutu, $[-16,+15]$ arama aralığı $45 çerçeve / saniye$ hız ve $720p$ HDTV çözünürlüğünde HK işlemi yapabilmektedir. İki mimarinin de benzer performansta çalışması, ikisi arasında bir seçim yapılmasına olanak tanıyan melez bir HK yongası tasarımını mümkün kılabilir.

Sentez sonuçlarına bakıldığında aynı HK yöntemini gerçekleyen iki farklı donanım mimarisi arasında bazı farklılıklar görülmektedir. Örneğin KPTDD ile tasarlanan Ç1BD temelli HK donanımı 16 adet Blok RAM içerirken, aynı HK yöntemini gerçekleyen HVTDD mimarisi sadece 4 adet Blok RAM içermektedir. Buna karşın HVTDD mimarisinde 72 adet daha fazla ROM kullanılmaktadır. Bu farklılık

incelendiğinde sentezleyicinin her iki mimaride de ortak olan İP'lerin içindeki LUT öbeklerinin sentezleyici tarafından farklı sentezlenmesinden kaynaklanmaktadır. Sentezleyici KPTDD mimarisinde buluna İP'lerin içindeki LUT öbeğini bir tane Blok RAM ile sentezlerken HVTDD mimarisinde bulunan İP'lerin içindeki LUT öbeğini 6 adet ROM kullanarak sentezlemiştir.

K-1BD yöntemi için elde edilen güç tüketimi değerleri Tablo 4.10'da görülmektedir. Burada da yine Ç1BD yöntemi ile karşılaştırma yapabilmek amacıyla iki mimarinin güç tüketimi sonuçları birlikte sunulmuştur. Tablo 4.10'daki değerler de yine "*mobile*" video dizisinin 100. ve 75. çerçevelerin aday çerçeve olarak seçilmesi yapılan HK işlemi sonucunda elde edilmiştir. Kullanılan HK yöntemleri farklı olduğu için Ç1BD yöntemi için elde edilen HV değerleri burada farklıdır. Post P&R işlemi yapılan donanımlar karşılaştırma amacıyla yine 66MHz çalışma frekansında test edilmiştir.

Tablodan da anlaşılacağı gibi K-1BD yönteminin daha fazla donanım kaynağı kullanmasından dolayı tasarlanan donanımın güç tüketimi, aynı mimarinin Ç1BD yöntemi için gerçeklenmesi sonucunda elde edilen güç tüketiminden yaklaşık 3 kat daha fazladır. Yine bu tabloda da HVTDD mimarisinin KPTDD mimarisine göre ortalama 2 kat daha fazla güç tüketimine neden olduğu açıkça görülmektedir. Ayrıca iki mimarinin bir arama penceresi için geçen sürede harcadıkları enerji arasında yaklaşık 4.4 kat fark vardır.

Tablo 4.10: K-1BD temelli HK yöntemi için elde edilen güç tüketimi sonuçları

Blok Konumu, Çerçeve Numarası	Hareket Vektörü	[48] (mW) HVTDD (66 MHz)	Önerilen (mW) KPTDD (66 MHz)
(15,11), 100	(0,0)	810	173
(12,10), 75	(0,0)	900	177
(12,12), 75	(1,0)	852	186
(12,8), 75	(-1,0)	855	200
(20,8), 75	(0,0)	975	286
(18,8), 100	(0,0)	777	155
Enerji		12960 mW.ns	2942 mW.ns

5. KESİK GRAY KODLANMIŞ BDU TEMELLİ HK YÖNTEMİ VE DONANIM MİMARİSİ

8-bit/piksel temelli HK mimarilerinin donanım karmaşasını artıran bir durum yüksek miktardaki aritmetik toplama işlemidir. Örneğin geleneksel 8-bit/piksel temelli İP mimarisinde 8'i mutlak fark işlemi, 16'sı da mutlak farkların toplanması olmak üzere toplam 24 tane tam toplayıcı bulunmaktadır. 1BD temelli bir İP mimarisinde bu sayı toplamda 15 tam toplayıcıdır. Bununla birlikte 8-bit/piksel temelli bir mimaride 1BD temelli HK yönteminde 1B bir sistolik dizi ile sağlanan performansı elde edebilmek için 2B bir İP dizisi kullanmak gerekmektedir. Dolayısıyla basit bir hesapla, 8bit/piksel temelli HK mimarisinin donanım karmaşasının tam toplayıcılar türünden ifade edilmesi gerekirse 6144 tam toplayıcıya gerek vardır. 1BD temelli HK donanımı mimarisinde ise 240 adet tam toplayıcıya gerek vardır. 1BD temelli HK donanımının bir dezavantajı, ikili video çerçevelerinin elde edilmesi için gerekli süzgeçleme işlemlerinin gerçeklenmesidir. 1BD yönteminde karşılaşılan bir başka durum da yöntemin doğasından kaynaklanan daha düşük HK başarımıdır.

Literatürde, düşük bit derinliğinde işlem yapmanın basitliğini ve 8bit/piksel yaklaşımlarının sahip olduğu yüksek HK doğruluğunu birleştirmek amacıyla bit kesme temelli HK mimarileri önerilmiştir [114,115,54]. [114] ve [115]'de, bit kesme temelli HK yöntemleri, VLSI mimarileri ile birlikte önerilmiştir. [54]'de sabit bit kesme algoritması ve uyarlanabilir bit maskeleme yönteminin birleştirmesiyle elde edilen yeni bir mimari önerilmiştir.

Literatürde HK işleminin hesapsal yükünü azaltmak için önerilen 1BD temelli çalışmalar ayrıntılı biçimde incelenmiş ve bu tez çalışması kapsamında bu yöntemler için önerilen mimari çözümler sunulmuştu. Bu bölümde, bit-kesme tekniğini kullanan düşük bit-derinliğine sahip yöntemler üzerine odaklanılmıştır. [114]'de piksel değerlerinin tüm bitlerini kullanmak yerine en değerlikli bitlerin birkaçının kullanılması ile daha hızlı bir HK işlemi gerçekleştirilmiştir. Benzer bir düşünceyle [115]'de bu işlemin uyarlanabilir bir şekilde yapılmasını öneren bir çalışma sunulmuştur. [116]'da bit kesme işlemi değişken blok boyutu için uygulanmıştır.

Bu tez çalışmasında, [114-116]'da önerilen klasik, doğrudan piksel değerlerinin ikili karşılığı üzerinden bit kesme yapan yöntemlerin aksine Gray kodlanmış bit uzaylarından kesme yaparak elde edilen bit-düzlemlerini kullanarak HK yapan kesik Gray kodlanmış BDU (KGKBDU) temelli HK yöntemi önerilmiştir. Bu yöntemin HK başarımının; 1BD, 2BD, Ç1BD ve K-1BD yöntemlerine göre daha yüksekt olduğu gösterilmiştir. Bunun yanında bu çalışmada önerilen yöntem ile dönüşüm işlemi klasik düşük bit derinliği gösterimi temelli HK yöntemlerine göre çok daha basit bir şekilde gerçekleştirilebilmektedir. Dolayısıyla, dönüşüm kısmı 1BD temelli yöntemlerden daha kolay ve 8 bit/piksel gösterimi temelli HK yöntemlerine kıyasla daha basit bir donanım mimarisi elde edilmesi mümkün olmaktadır.

Bu bölümde önerilen özgün yöntem için tasarlanan donanım mimarisi de tez kapsamında geliştirilmiş 1BD temelli HK mimarilerde olduğu gibi KPTDD mimarisi temelinde tasarlanmıştır. Ancak HK yönteminin yapısından kaynaklanan küçük mimari değişiklikler yapılmıştır.

5.1 KGKBDU Temelli HK Yöntemi

[117]'de HK işleminin yükünü, özellikle donanım tarafında azaltmak amacıyla Gray kodlama temelli bir HK yöntemi önerilmiştir. Bu bölümde [117]'de önerilen çalışmada bir değişiklik yapılarak gerçekleştirilen, KGKBDU temelli HK işleminde elde edilebilecek başarım ve bunun sonucunda donanım tarafında elde edilebilecek sadeleşme olasılığı irdelenmiştir. [117]'deki çalışmada Gray kodlanmış video çerçevelerinde piksellerin bütün bitleri birbirleri kullanılarak uyumlama işlemi yapılırken bu tez çalışmasında önerilen yöntemde piksellerin Gray kodlanmış ikili gösterimlerinde, HK başarımı çok fazla etkilenmeyecek şekilde, bit-kesme tekniği kullanılarak basitleştirme yapılmıştır.

t çerçevesinde, (i,j) konumundaki pikselin 2^K adet gri seviyede nicemlenmesi ile piksel değeri için (5.1)'deki ifade elde edilir:

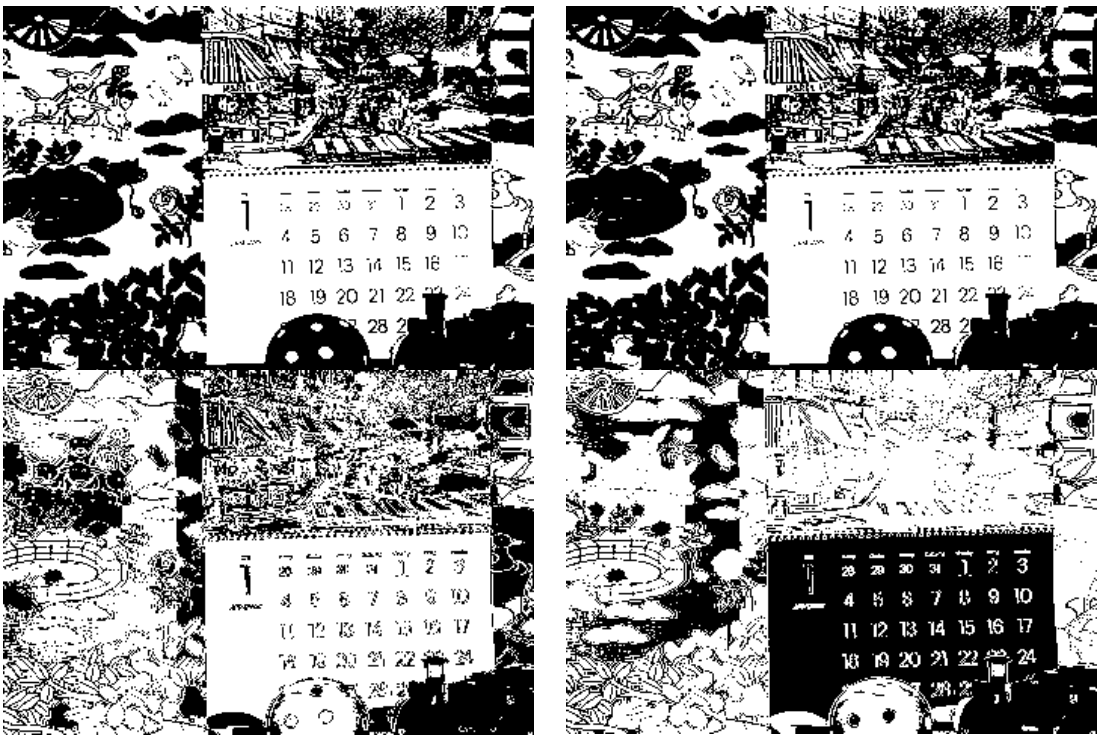
$$I^t(i,j) = a_{K-1}2^{K-1} + a_{K-2}2^{K-2} + \ldots + a_1 2^1 + a_0 2^0 \qquad (5.1)$$

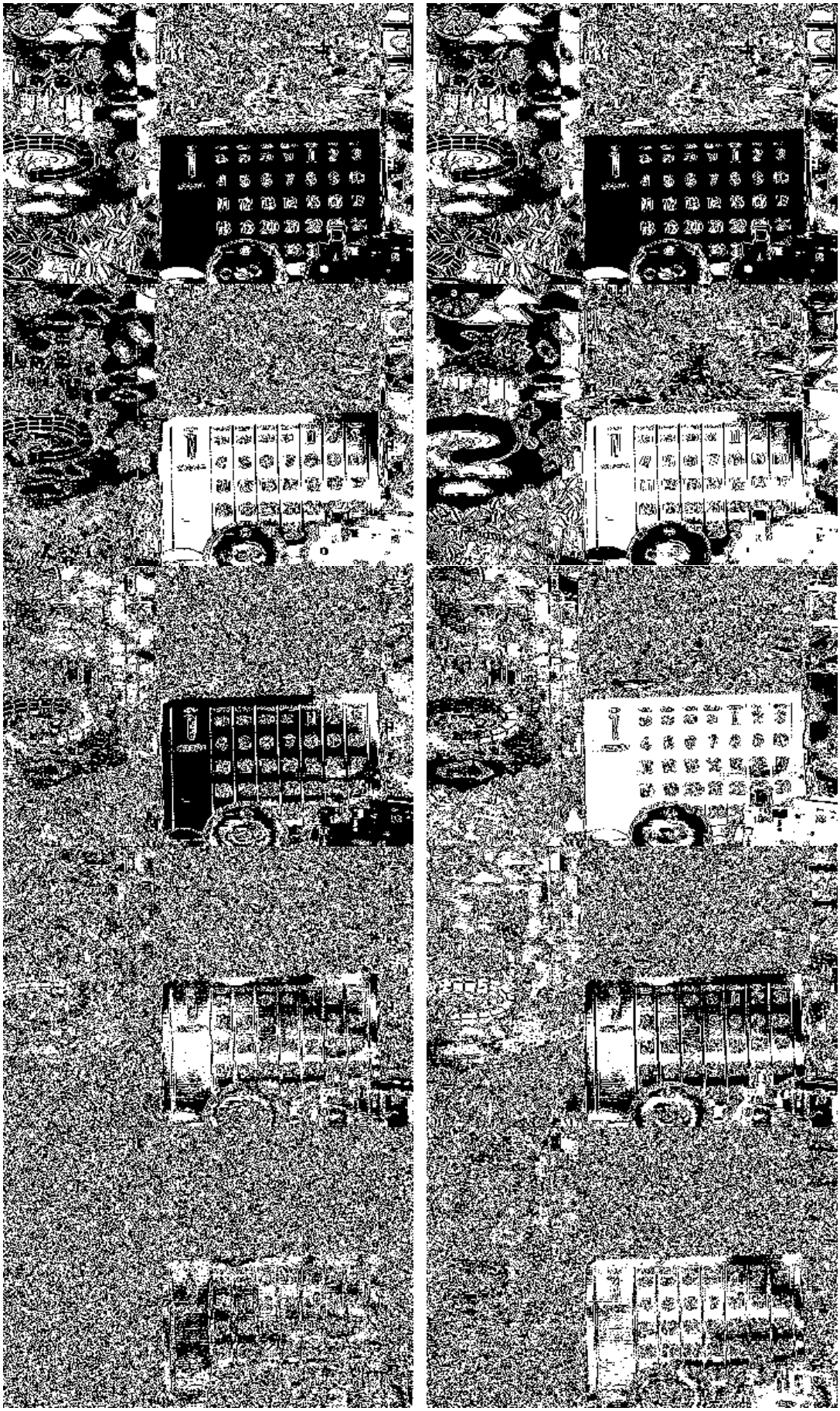
(5.1)' de, a_k ikili kodlardır ve sadece ikili değerler alırlar. Şayet t. çerçevenin k. bit düzlemi $b_k^t(i,j)$ şeklinde ifade edilecek olursa bu düzlem k. seviyedeki bütün a_k bitlerini içerir. Burada K'nın değeri 8 olarak seçilecek olursa $b_0^t(i,j)$ en değerliksiz bit düzlemini, $b_7^t(i,j)$ ise en değerlikli bit düzlemini ifade etmektedir.

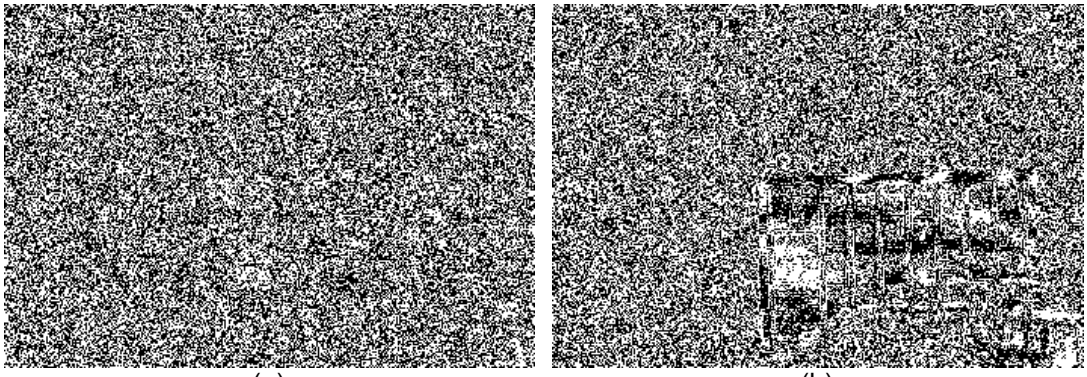
Bir pikselin Gray kodlanmış halini elde etmek için pikselin kendi değeri kullanılarak yapılan işlem (5.2)'deki gibidir:

$$\begin{aligned} g_{K-1} &= a_{K-1} \\ g_k &= a_k \oplus a_{k+1} \end{aligned}, \quad 0 \le k \le K-2 \tag{5.2}$$

Burada "$\oplus$", XOR işlemini göstermektedir. Gray kodlanmış piksellerde komşu seviyeler arasında sadece bir bit değerini değiştirmektedir. Bit düzlemi uyumlaması temelli HK yöntemi için Gray kodlanmış piksel değerlerinin kullanılması bu nedenle uyum işlevinin daha sağlıklı sonuçlar üretmesini sağlamaktadır. "*mobile*" video dizisinin birinci çerçevesinin kendisinin ve gray kodlanmış halinin 8 ayrı bit düzlemi ifadesi Şekil 5.1'de görülmektedir. Yukarıdan aşağıya doğru bit düzlemleri en değerlikliden en değerliksiz bite doğru olacak şekilde sıralanmışlardır.







(a) (b)

Şekil 5.1: "*mobile*" video dizisinden alınan bir çerçevesindeki piksellerin 8 ayrı bit düzlemi şeklindeki gösterimi: a) Normal piksellerin bit düzlemleri, b) Gray kodlanmış piksellerin bit düzlemleri

Gray kodlama temelli HK işleminde benzerlik, *ilişki ölçüsü (CM$_G$)* ile gösterilmektedir. *CM$_G$* ölçütü (5.3)'deki ifadede görülmektedir. Bu ifadede de (5.1)'deki gibi her bir ikili sayı, sahip olduğu değerliğe bağlı olarak ağırlıklandırılmıştır. Şekil 5.1'de görülen bit düzlemlerine bakıldığında Gray kodlanmış bit düzlemleri ile normal bit düzlemlerinin içerdikleri detay bilgilerinin yakın sevilerde olduğu görülmektedir. Dolayısıyla (5.3)'deki gibi bir ağırlaklındırma yanlış olmayacaktır.

$$CM_G(m,n) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\sum_{k=0}^{K-1} 2^k \times \left\{ g_k^t(i,j) \oplus g_k^{t-1}(i+m, j+n) \right\}, \quad -s \le m,n \le s-1 \tag{5.3}$$

Burada (m,n) ve s sırasıyla aday yer değiştirmesini ve arama aralığını göstermektedir. En düşük ilişki değerine sahip yer değiştirme değeri ilgili bloğun hareket vektörü olarak atanmaktadır. Yüksek değerlikli bitlerin daha büyük değerleri ifade ettiği göz önünde bulundurularak, ilişkinin hesaplanmasında 2^k şeklinde bir ölçekleme değişkeni kullanılmaktadır.

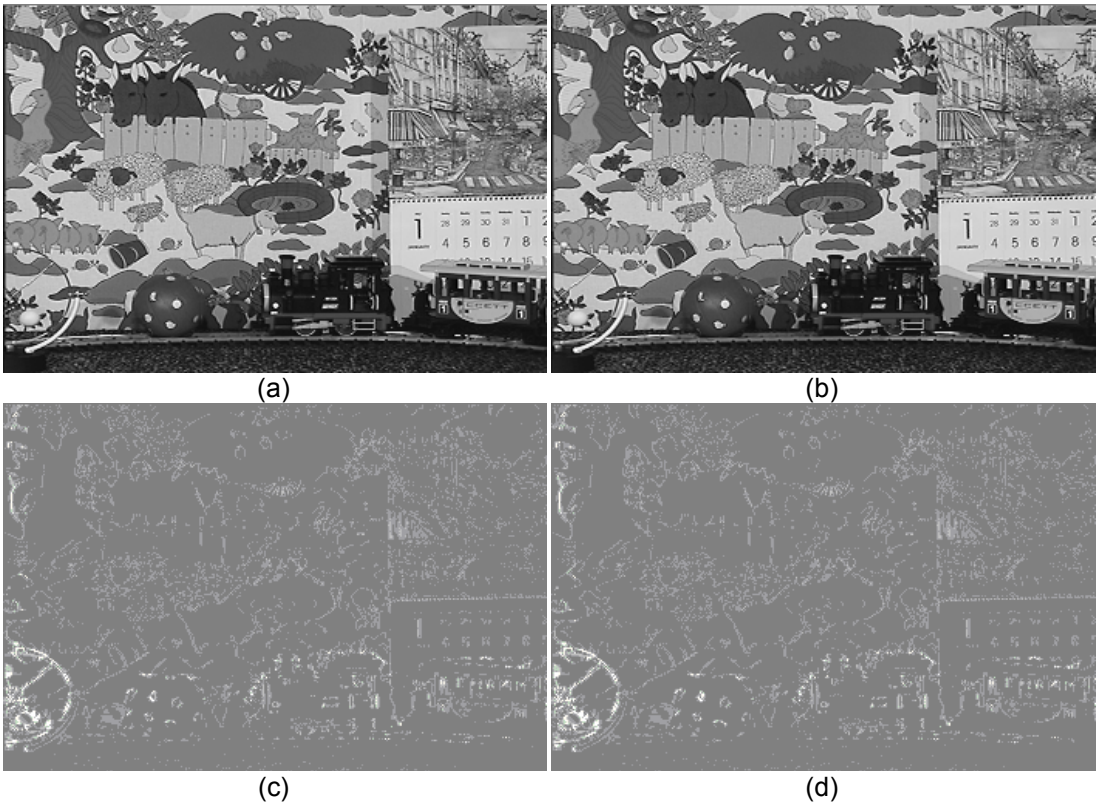
Önerilen Gray kodu üzerinden bit kesme yönteminde HK için, *CM$_G$* hesaplanırken en yüksek değerlikli K tane bit kullanılmaktadır. Kesilen bitlerin sayısı NTB (Number of Truncated Bits) ile gösterilirse uyumlama için $M = K - NTB$ adet bit düzlemine gerek vardır. Bu durumda uyulmama ölçütü (5.4) şeklini almaktadır.

$$CM_G(m,n) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\sum_{k=NTB}^{K-1} 2^{k-NTB} \times \left\{ \begin{matrix} g_k^t(i,j) \\ \oplus g_k^{t-1}(i+m, j+n) \end{matrix} \right\}, \quad (5.4)$$

$$-s \le m,n \le s-1$$

Dikkat edilirse benzerlik ölçütünün hesabı için (5.4)'de verilen denklemdeki toplama işlemlerinin sadece alt sınırı değişmektedir, ancak bu küçük değişim hesapsal yükün azaltılmasında çok büyük bir rol oynamaktadır. Burada ayrıca dikkat edilmesi gereken bir nokta da bütün işlemlerin tamamen ikili gerçekleştirilebilir olmasıdır. Örneğin, çarpma işlemi ikinin kuvveti olduğundan, basit bir kaydırma işlemi şeklinde gerçekleştirilebilir.

Deneysel sonuçlar göstermiştir ki, *NTB=5* değeri hesapsal yük ile HK doğruluğu arasında donanım karmaşası da göz önüne alındığında iyi bir denge oluşturmaktadır. Dolayısıyla, önerilen HK yönteminin donanım tasarımı sadece üç adet bit düzlemi kullanılacak şekilde yapılmıştır. Farklı *NTB* değerleri için elde edilen HK başarımına ait deneysel sonuçlar bölüm sonunda ayrıca sunulmuştur. Şekil 5.2'de MFT kullanılarak yapılan HK ve KGKBDU kullanılarak yapılan HK sonucu yeniden oluşturulan güncel video çerçeveleri ve bu çerçevelerin gerçek güncel video çerçevesi ile arasındaki fark görülmektedir.

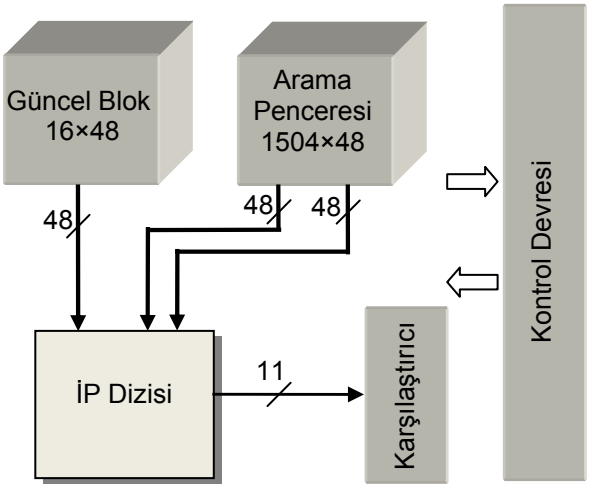


Şekil 5.2: KGKBDU yönteminin "*mobile*" dizisinden bir çerçeve üzerinde uygulanması ile elde edilen sonuçlar a)Video çerçevesinin MFT ile HK yapıldıktan sonra yeniden oluşturulmuş görünümü, b) Video çerçevesinin KGKBDU yöntemi ile HK yapıldıktan sonra yeniden oluşturulmuş görünümü, c) Gerçek video çerçevesi ile (a)'da görülen yeniden oluşturulmuş video çerçevesi arasındaki fark, d) Gerçek video çerçevesi ile (b)'de görülen yeniden oluşturulmuş video çerçevesi arasındaki fark

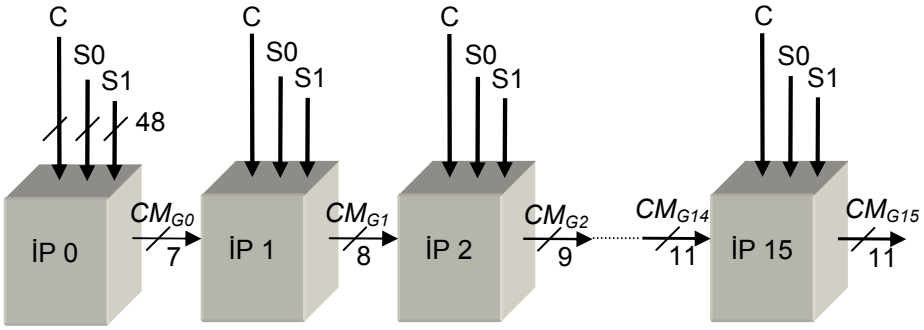
5.2 KGKBDU Temelli HK Donanımı Mimarisi

Bu kısımda, önerilen HK yöntemi için 1B sistolik dizi tasarlanmıştır. Şekil 5.3'de, görülen donanım; [-16, 15] arama aralığında makro-blok seviyesinde HK yapabilmektedir. Güncel blok belleği 48 bit genişliğinde ve 16 satır derinliğinde bir RAM öbeği ile gerçeklenmiştir. Arama penceresi yine 48 bit genişliğinde ve 1504 satır derinliğinde iki çıkışlı bir RAM öbeği ile gerçeklenmiştir. Arama penceresi için akıllı bir okuma devresi tasarlanarak ihtiyaç duyulan yonga üzeri bellek miktarı azaltılabilir, fakat bu durum tez kapsamında yürütülen çalışmada ele alınmamıştır.

Teorik olarak, önerilen HK mimarisinde arama penceresi için gereken en düşük bellek alanı 3-bit/piksel×47×47= 6,627k bit dir. Böylelikle, yonga üzeri bellek ihtiyacını toplamda 7,395k bite düşebilmektedir. Ancak donanımın tasarımında bu kısma girilmemiş ve (1504+16)×48 = 72,96k bit bellek kullanılarak bu ihtiyaç karşılanmıştır.



Şekil 5.3: Önerilen yöntemin donanım mimarisi



Şekil 5.4: İP Dizisi

1BD temelli HK mimarilerinde 16 bitlik satır vektörleri tek bir İP de işlenmektedir. Bellek bu yapıya göre düzenlendiği için arama penceresindeki 1 adımlık kayma için önceden okunmuş olan 16 bitlik vektörün arama yönü doğrultusundaki 15 biti, yeniden okunmak durumunda kalınmaktadır. Burada, belleğe yerleştirilen verilerin bir şekilde bölünerek akıllı bir okuma devresi tasarımı gerekmektedir. Bu tasarımın getirileri ve götürüleri arasındaki ödünleşimin iyi yapılması ve buna göre bir sonraki adıma karar verilmesi daha uygun olacaktır. [118]'de önerilen 1BD temelli HK mimarisinde toplam yonga üzeri bellek miktarı 24,32k bit dir. Diğer taraftan [9]'da önerilen 8-bit/piksel temelli HK mimarisinde ise toplam yonga üzeri bellek alanı 208k bit dir.

Şekil 5.4'de görülen yapı, 16 İP öbeğinin yarı sistolik bir dizi şeklinde yerleştirilmesi ile oluşmuştur. Şekilden de anlaşılacağı gibi $İP_0$ biriminin CM_{Go} girişi yoktur çünkü ilk CM_{Go} değeri bu öbeğin içinde hesaplanacaktır. Bütün İP'ler büyük oranda birbirleriyle özdeştirler. Farklılık, her bir İP'deki CM_G giriş ve çıkışlarının genişliğinde, dolayısıyla bu girişlerin hesabında kullanılan aritmetik yapılardadır. Her bir İP'nin CM_G çıkışı, öbek içerisindeki toplayıcı ağacının çıkışı ile eğer varsa CM_G girişinin

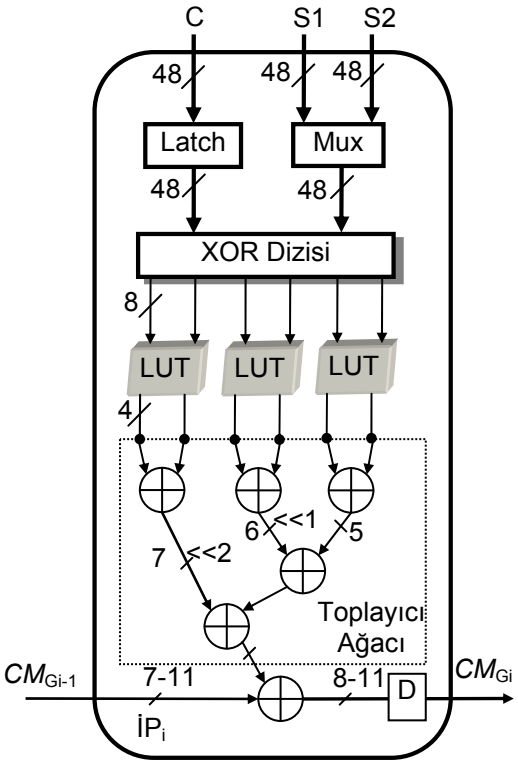
toplanması ile elde edilir. Bu durum Şekil 5.4'de görsel olarak ifade edilmektedir. Şekilden de anlaşılacağı gibi CM_{G0}, CM_{G1}, ve son olarak CM_{G15} sırasıyla 7, 8 ve 11 bit genişliğindedir. Bu farklılıklar donanımın tasarımında hesaba katılmıştır.

Her bir İP de toplam 23 tam toplayıcı olduğu ve CM_G ölçütünün hesabında 153 tam toplayıcıya ihtiyaç olduğu düşünüldüğünde, İP dizisinde ihtiyaç duyulan toplam tam toplayıcı sayısı 523 dür. [118]'de önerilen çalışmada bu sayı 192 dir ancak bunun karşılığında HK doğruluğu düşmektedir. Diğer taraftan [9]'da önerilen çalışmada 4×4 boyutlu bir blok için gerekli SAD değerinin hesabında kullanılan İP dizisindeki toplam tam toplayıcı sayısı 460'dır. Bu durumda [118]'deki temel alınarak 16×16 boyutlu bir blok için gerekli donanımda, 7360 tam toplayıcı kullanılması gerekecektir. Sonuç olarak önerilen yöntem için tasarlanan donanımın karmaşıklığı/büyüklüğü 1BD ve 8-bit/piksel temelli HK mimarisinin arasındadır ancak 1BD temelli mimarilere daha yakındır.

Önerilen yöntem için tasarlanan donanımda kullanılan İP mimarisi Şekil 5.5'de görülmektedir. CM_G girişleri dışında her bir İP'de üç giriş daha vardır ve bu girişlerden bir tanesi güncel blok diğer ikisi ise aday blok içindir.

1BD temelli mimarilerde UNS ölçütünü hesaplamak için LUT kullanılmaktadır. Bu bölümde önerilen mimarideki İP'de, kullanılan LUT sayısı üçe çıkarılmıştır çünkü 3 adet bit düzlemi uyumlanmaktadır.

Güncel blok belleğinden okunan veriler İP'ler arasında kaydırılmak yerine her bir İP'ye bu belleğin tek bir satırı gitmektedir ve bu sadece bir kez yapılıp diğer durumlarda bu veri İP içerisindeki *Latch* devresinde tutulmaktadır. Bu durum daha detaylı bir biçimde Tablo 5.1'de görülen veri akış yapısında verilmiştir.



Şekil 5.5: Önerilen donanımda bulunan İP öbeğinin iç yapısı

Tablo 5.1: Önerilen mimarinin sahip olduğu ver akışı yapısı

Çevrim Zamanı	Giriş Verisi			İşlemci Girişleri					Hesaplanan Blok Bozlumasının Ait Olduğu Konum
	C	*S1*	*S2*	İP-0	İP-1	. . .	İP-14	İP-15	
0	C0	S0,0		C0-S0,0					
1	C1	S1,0		[C0]-S1,0	C1-S1,0				
2	C2	S2,0		[C0]-S2,0	[C1]-S2,0				
.	.	.		.	.				
.	.	.		.	.				
.	.	.		.	.				.
.	.	.		.	.	…			.
14	C14	S14,0		[C0]-S14,0	[C1]-S14,0	…	C14-S14,0		.
15***	C15	S15,0		[C0]-S15,0	[C1]-S15,0	…	[C14]-S15,0	C15-S15,0	D(0,0)
16	C0	S16,0		[C0]-S16,0	[C1]-S16,0	…	[C14]-S16,0	[C15]-S16,0	D(1,0)
.	.	.		.	.	.	.	.	.
.	.	.		.	.	.	.	.	.
.	.	.		.	.	.	.	.	.
31	C15	S31,0		[C0]-S31,0	[C1]-S31,0	…	[C14]-S31,0	[C15]-S31-0	D(16,0)
32	C0	S32,0	S0,3	[C0]-S0,3	[C1]-S32,0	…	[C14],S32,0	[C15]-S32,0	D(17,0)
.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.
46	C14	S46,0	S14,3	[C0]-S14,3	[C1]-S14,3	…	[C14]-S14,3	[C15]-S46,0	D(31,0)
47	R15		S15,3	[C0]-S15,3	[C1]-S15,3	…	[C14]-S15,3	[C15]-S15,3	D(0,1)
***Bu andan itibaren bütün PE' ler çalışmaktadır									

Tablo 5.1'de köşeli parantez içinde görülen veriler RAM bellek yerine İP içerisindeki *Latch* öbeğinden okunan verilerdir. Sonuç olarak bir MB konumu için yapılan HK işleminde, her bir İP için güncel blok belleğinden sadece bir kez okuma yapılmaktadır. Tabloda görülen C0, güncel bloğun birinci satırındaki 48 bit

genişliğindeki satır vektörüdür. Bu vektör 3 bit genişliğindeki 16 pikselin yan yana birleştirilmesiyle elde edilmiştir. Si,j terimi ise arama penceresinin *i.* satırında *j* ve *(j+47)* sütunlarında bulunan 48 bitlik satır vektörünü temsil etmektedir. Sonuç olarak tasarlanan donanım bir saat çevriminde 16 piksel işleyebilmektedir. Tablo 5.1'den görüleceği üzere tüm İP'lerin çalışır duruma gelebilmesi için 15 saat darbesine ihtiyaç vardır. Tasarlanan donanım bir MB için hareket vektörünü 1024 çevrimde hesaplar. Başlangıçta gereken 15 çevrimde hesaba katıldığında toplam 1039 çevrime ihtiyaç vardır.

İP mimarisinde görülen *Mux* öbeği Tablo 5.1'de görülen veri akışı yapısı doğrultusunda üretilen denetim sinyaline bağlı olarak arama penceresi belleğinden gelen verilerin seçilmesi işlemini gerçekleştirir. *Latch* ve *Mux* öbeklerinin çıkışları XOR dizisine gönderilir ve burada karşılıklı pikseller arasındaki uyum ölçülür. XOR öbeğinin çıkışı üç guruba ayrılarak eşit değerlikli bitler bir araya toplanır. Her bir piksel 3 bit ile ifade edildiği için üç adet 16 bitlik vektör elde edilir ve bu vektör İP içerisindeki üç ayrı LUT öbeğinin girişine uygulanır.

LUT çıkışları toplayıcı ağacına girerek ağırlıkları ile orantılı olarak bir birbirleriyle toplanırlar. Toplayıcı ağacının çıkışında en kötü durumda 7 bit genişliğinde bir sayı olacaktır. Son olarak bir önceki bloktan gelen CM_G girişi ile birlikte toplayıcı ağacının çıkışı toplanarak, bir sonraki İP ye giriş olarak gidecek CM_G çıkışı hesaplanır.

[118]'de sunulan mimari ile karşılaştırma yapmak amacıyla Xilinx XST aracı kullanılarak örnek bir sentez işlemi yapılmıştır. Sentez sonuçları ve Post P&R sonuçları Tablo 5.2'de görülmektedir.

Tablo 5.2: Sentez ve Post P&R sonuçları.

HK Yöntemi	KGK1BD
Mimari	KPTDD
Frekans (MHz)(Sentez)	90
Frekans (MHz)(Post P&R)	73
Kullanılan LUT sayısı Yonga yararlanımı	2339 (%8)
Kullanılan İki uçlu RAM sayısı	96

Sentez işlemi sonucunda tasarlanan donanım, XC2VP30 FPGA yongasında 2339 CLB(Configurable Logic Block)'lik bir alan kaplamıştır. Aynı FPGA yongası için [118]'de önerilen çalışma Ç1BD yöntemi için tasarlanan mimari 1121, K-1BD

yöntemi için tasarlanan mimarisi ise 1721 adet LUT ile gerçeklenmiştir. Önerilen mimari alan bakımından [118]'de önerilen Ç1BD mimarisine göre yaklaşık iki kat daha büyük görünmektedir buna karşılık piksel bit derinliği 3 kat daha fazla, dolayısıyla elde edilen HK doğruluğu daha yüksektir. Donanımın kapladığı alanın Ç1BD donanımı mimarisinin 3 katından az olmasında, CM_G değerlerinin toplanması için gerekli tasarımların daha dikkatli yapılması gibi etkenler rol oynamıştır.

Tablo 5.3'de önerilen donanım mimarisinin giriş verisi olarak sırayla, Gray kodlanmış bit düzlemlerini ve normal bit düzlemlerinin kullanılması ile elde edilen güç tüketimi sonuçları görülmektedir. Testler 66MHz çalışma frekansı üzerinde ve "*mobile*" dizisinin 75. çerçevesi kullanılarak gerçekleştirilmiştir. Tablo 5.3'de görüldüğü üzere Gray kodlanmış bit düzlemleri kullanılarak yapılan HK işlemi, tasarlanan donanımın, test edilen bloklar için ortalamada, yaklaşık %6.73 daha az güç tüketmesini sağlamaktadır.

Tablo 5.3: K-1BD temelli HK yöntemi için elde edilen güç tüketimi sonuçları

Blok Konumu	Gray Kodlanmış Bit Düzlemleri (66MHz)		Normal Kodlanmış Bit Düzlemleri (66MHz)	
	Güç Tüketimi(mW)	Hareket Vektörü	Güç Tüketimi(mW)	Hareket Vektörü
(2,2)	230	(-1,0)	240	(-1,0)
(2,6)	196	(-1,0)	221	(-1,0)
(2,10)	207	(0,0)	233	(0,0)
(2,14)	236	(-1,0)	260	(-1,0)
(12,2)	260	(0,1)	289	(-1,1)
(12,6)	249	(0,0)	261	(0,0)
(12,10)	183	(0,0)	186	(0,0)
(12,14)	229	(1,-1)	244	(1,-1)
(21,2)	260	(0,0)	286	(0,0)
(21,6)	252	(0,0)	264	(0,0)
(21,10)	208	(0,0)	210	(0,0)
(21,14)	223	(1,-1)	225	(1,0)
Enerji	4140 mW.ns		4422 mW.ns	

5.3 KGKBDU Temelli HK Yönteminin Geçmişte Önerilen Benzer HK Yöntemleri ile Karşılaştırılması

Deneysel sonuçların elde edilmesi aşamasında, önerilen KGKBDU yaklaşımı açık çevrim yöntemi ile test edilmiştir. Burada güncel çerçeve bir önceki referans çerçeve üzerinde HK yapıldıktan sonra elde edilen hareket vektörleri kullanılarak geri çatılmıştır. Daha sonra geri çatılan imge ile güncel imgenin farkları alınarak PSNR ölçütü ile başarım değerlendirilmiştir.

Deneysel sonuçların elde edilmesi aşamasında, önerilen yöntemin başarımını sağlıklı değerlendirebilmek için altı farklı video dizisi kullanılmıştır. Bu diziler için elde edilen ortalama PSNR değerleri Tablo 5.4'de görülmektedir. Tabloda görülen T-BPM değerleri [114]'de sunulan klasik bit kesme yöntemini belirtmektedir. *NTB,* kesilen bit sayısını göstermektedir. *NTB* değeri 7 örneğin sadece en değerlikli bitin kullanılacağını göstermektedir.

Tablo 5.4'den görüldüğü üzere, K-1BD ve 2BD gibi klasik düşük bit gösterimi temelli HK yöntemlerinin klasik bit kesme yöntemine göre daha başarılı sonuçlar vermektedir. Bunun yanında, önerilen yöntem Gray kodlama sayesinde klasik bit kesme yönteminden, her durumda daha iyi sonuç vermektedir. Ayrıca 3 bit kullanıldığı durumlarda önerilen yöntem, iki-bit düzlemi kullanılan yöntemlere göre daha iyi sonuçlar vermektedir.

Tablo 5.4: Hareket Vektörleri Kullanılarak Geri Çatılan İmge Dizilerine Ait Ortalama PSNR Değerleri

	Video Dizileri					
Yöntem	"football" 352x240 125	"flowergarden" 352x240 115	"mobile" 352x240 140	"tennis" 352x240 112	"coastguard" 352x288 300	"foreman" 352x288 300
MFT	22.88	23.79	22.99	29.87	30.48	32.11
1BD [48]	21.83	23.32	22.71	28.77	29.84	30.44
2BD [50]	22.08	23.43	22.72	28.89	29.93	30.71
Ç1BD [51]	21.81	23.26	22.73	28.78	29.88	30.38
K-1BD [52]	22.10	23.39	22.77	29.18	29.98	30.87
KBDU (*NTB*=6)	22.08	23.49	22.72	28.57	28.97	30.35
KBDU (*NTB*=5)	22.22	23.49	22.75	28.68	29.85	30.86
KBDU (*NTB*=4)	22.22	23.49	22.76	28.70	29.95	31.04
KBDU (*NTB*=3)	22.21	23.48	22.76	28.70	29.95	31.08
KBDU (*NTB*=2)	22.20	23.48	22.76	28.70	29.95	31.09
KGKBDU (*NTB*=6)	22.39	23.61	22.84	28.98	29.14	30.64
KGKBDU *(NTB=5)*	*22.59*	*23.67*	*22.86*	*29.19*	*30.16*	*31.32*
KGKBDU (*NTB*=4)	22.58	23.66	22.87	29.26	30.27	31.57
KGKBDU (*NTB*=3)	22.56	23.66	22.87	29.23	30.26	31.61
KGKBDU (*NTB*=2)	22.56	23.66	22.87	29.23	30.25	31.61
AKGKBDU (NTB=6)	22.40	23.57	22.79	29.15	29.16	30.71
AKGKBDU (NTB=5)	22.46	23.58	22.78	29.40	30.13	31.37
AKGKBDU (NTB=4)	22.23	23.47	22.75	29.31	30.10	31.39
AKGKBDU (NTB=3)	22.05	23.38	22.71	28.99	29.94	31.21
AKGKBDU (NTB=2)	21.94	23.34	22.68	28.84	29.84	31.05

Tablo 5.4'de AKGBDU ile gösterilen sonuçlar denklem (5.3) ve (5.4)'de görülen ağırlıklandırma faktörlerinin kaldırılması ile elde edilmiştir. Ağırlıklandırma faktörünün kullanılması genel olarak HK başarımını bir miktar arttırmaktadır.

Bu bölümde gray kodlanmış bit düzlemleri üzerinde yapılan kesme işlemine dayalı özgün bir HK yöntemi ve donanım mimarisi önerilmiştir. Önerilen yöntem, geçmişteki düşük bit gösterimi temelli HK yöntemlerinden daha iyi sonuçlar vermiştir. Önerilen yöntem ayrıca klasik bit kesme temelli HK yöntemlerinden de daha iyi bir HK başarımına sahiptir. Önerilen yöntem için ayrıca etkin bir donanım mimarisi de önerilmiş ve gerçeklenmiştir. Önerilen HK yöntemi ve donanım mimarisi birçok tüketici elektroniği uygulaması için etkin bir çözümdür.

6. Ç1BD TEMELLİ KESİRLİ HAREKET KESTİRİMİ YÖNTEMİ VE DONANIM MİMARİSİ

Bu bölümde güncel video kodlama yöntemlerinden H.264'ün önemli bileşenlerinden bir tanesi olan KHK üzerine yapılan araştırmalar sonucu önerilen 1BD temelli KHK yöntemi ve özgün donanım mimarisi anlatılmaktadır. Önerilen KHK yönteminin kazandırdıkları ve kaybettirdikleri detaylı olarak incelenmiştir. Yine bu çalışmaya özgü olan KHK mimarisinin geleneksel 1BD temelli tamsayı HK (THK) mimarisinin çalışmasına ne kadar yük getirdiği bunun yanında elde edilen yenilikler ve diğer 8-bit/piksel gösterimi temelli mimarilere göre getirdiği kolaylıklar açıklanmıştır.

KHK işleminin 1BD temelli bir yönteme uyarlanması aşamasında, getirdiği basitlik açısından Ç1BD yöntemi seçilmiştir. Bu yöntem kullanılarak çeyrek piksel doğrulukta HK yapabilen ve 8-bit/piksel temelli KHK yöntemlerine göre hesap yükü büyük oranda düşük olan bir KHK yöntemi geliştirilmiştir. Önerilen mimaride gerekli THK yapısı için önceki bölümlerde tasarlanmış olan Ç1BD yapısı kullanılmıştır. Literatürde 1BD temelli KHK işlemi sadece [119]'da ele alınmıştır. [119]'da, video çerçevelerinin çözünürlüğünü artırmak için 8-bit/piksel video çerçeveleri kullanılarak elde edilen yeni üst örneklenmiş video çerçevelerine 1BD dönüşümü uyguladıktan sonra arama işlemi gerçekleştirilmiştir. Bu işlem geleneksel KHK yaklaşımında olduğu gibi THK işleminin işaret ettiği noktanın etrafında değil, bütün yarım piksel aday konumları için yarım piksellik adımlarla tam arama işlemi yapılmıştır. Bu tez çalışmasında önerilen KHK yönteminde ise bütün işlemler 1BD gerçekleştirildikten sonra yapılmaktadır. Önerilen yöntem ayrıca, güncel video kodlama standardı olan H.264/AVC üzerinde test edilmiştir.

6.1 Ç1BD Temelli KHK Yöntemi

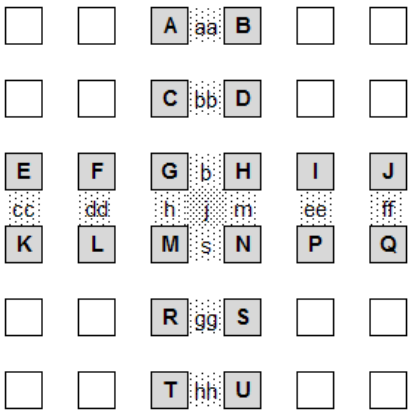
KHK, H.264/AVC gibi güncel video kodlama standartlarının sahip olduğu önemli bir özelliktir. KHK, yani yarım piksel ve çeyrek piksel HK, THK kestirimi işleminin başarımını artırmak için kullanılır. KHK işlemindeki ilk aşama tipik olarak aradeğerleme işlemidir.

Yarım piksel HK yapmak için gerekli olan aradeğerleme işlemi güncel video çerçevesi kullanılarak gerçekleştirilir. Her bir yarım piksel, altı komşu tamsayı piksel kullanılarak altı adımlı sonlu dürtü yanıtlı (FIR) bir süzgeç ile gerçekleştirilen aradeğerleme işlemi sonucunda elde edilir.

Şekil 6.1'de örnek bir yarım piksel aradeğerleme işlemi görülmektedir. Örneğin bu şekilde görülen **b** pikseli aşağıdaki gibi hesaplanır.

$$\mathbf{b} = round\left(\left(E - 5F + 20G + 20H - 5I + J\right)/32\right) \tag{6.1}$$

Bu tez çalışmasında önerilen Ç1BD temelli KHK yönteminde kullanılan yarım piksel aradeğerleme işlemi (6.1)'deki ile aynıdır ancak 8 bitlik pikseller yerine 1 bitlik pikseller üzerinden yapılmaktadır. Dolayısıyla bu şekilde gerçekleştirilen aradeğerleme işlemi daha basittir.

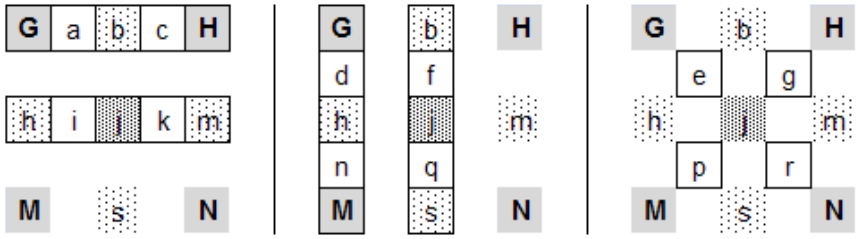


Şekil 6.1: Yarım piksel aradeğerleme örneği

Yarım piksel ara değerleme işlemi yapıldıktan sonra geleneksel KHK yöntemlerinde izlenen adım, tamsayı hareket vektörünün işaret ettiği tamsayı konumun etrafındaki olası sekiz adet yarım piksel konumunda tam arama işlemi yapmak sureti ile yarım piksel doğruluğundaki hareket vektörünün bulunmasıdır.

Yarım piksel doğruluktaki hareket vektörü bulunduğunda sıra aynı işlemin çeyrek piksel doğrulukta gerçekleştirilmesidir. Bu işlem için öncelikle çeyrek piksellerin elde edilmesi gerekmektedir. Çeyrek pikselleri elde etmek için yine aradeğerleme işlemine başvurulur ancak bu kez yarım piksel aradeğerleme işlemindekine kıyasla sadece değeri hesaplanacak çeyrek pikselin komşusu iki tam/yarım pikselin aritmetik ortalaması alınarak, daha düşük dereceden bir aradeğerleme işlemi yapılır.

Çeyrek piksel aradeğerleme işlemini gerçekleştirmek için göz önünde bulundurulması gereken üç farklı durum Şekil 6.2'de görülmektedir.



Şekil 6.2: Çeyrek piksel aradeğerleme a) Yatay doğrultuda b) Düşey doğrultuda c) Köşegen doğrultusunda

H.264/AVC standardında yatay ve düşey konumdaki çeyrek pikselleri hesaplamak için bazı tamsayı pikseller ve bazı yarım pikseller kullanılmaktadır, ancak köşegenler üzerinde bulunan çeyrek piksellerin hesaplanması için sadece yarım piksellerin kullanılması gerekmektedir. Çeyrek piksel aradeğerleme işlemi basit bir doğrusal aradeğerleme işlemidir ve örneğin (6.2)'de ifade edilmektedir.

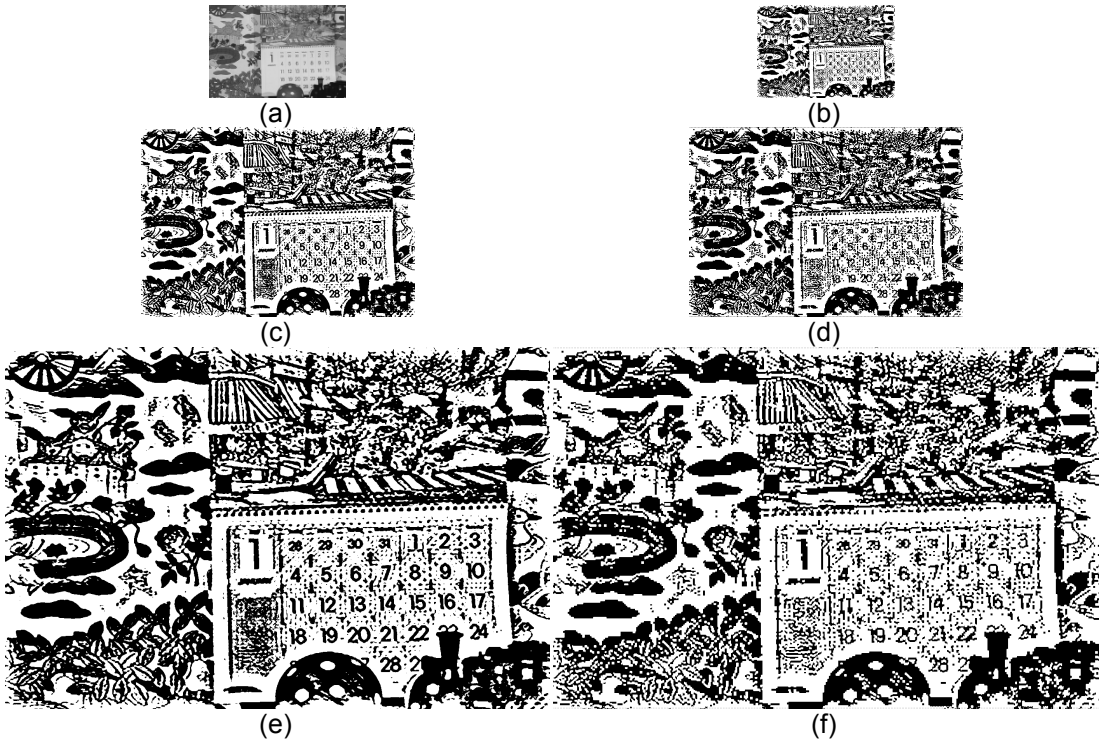
$$\mathbf{a} = round\left(\left(G + \mathbf{b}\right)/2\right) \tag{6.2}$$

Bu tez çalışması kapsamında, Ç1BD temelli KHK yönteminde bu aritmetik işlem basit mantıksal VEYA işlemine dönüştürülmüştür, bu şekilde ikili çeyrek piksel aradeğerleme denklemi aşağıdaki şekli almıştır. (6.3)'de görülen ||, mantıksal VEYA işlemini temsil etmektedir. Mantıksal VEYA işlemi aritmetik toplama işleminin ikili seviyede ifadesi olarak kullanılmıştır. Dolayısıyla bu işlemi gerçekleyecek donanım mimarisinde büyük oranda bir sadelik elde edilebilmektedir.

$$\mathbf{a} = G \,||\, \mathbf{b} \tag{6.3}$$

Aradeğerleme işleminin 8bit/piksel video çerçeveleri üzerinden yapıldığı [119]'da önerilen KHK yöntemi ile bu tez çalışmasında önerilen Ç1BD temelli KHK yöntemi ile gerçekleştirilen aradeğerleme işlemi sonucunda elde edilen ikili imgeler Şekil 6.3'de görülmektedir.

Şekil 6.3(a)'da *"mobile"* dizisinden örnek bir çerçeve görülmektedir ve Şekil 6.3(b)'de aynı imge üzerinde Ç1BD yöntemi kullanılarak elde edilen ikili imge görülmektedir. Şekil 6.3(c) ve Şekil 6.3(d)'de sırasıyla [119] ve bu tez çalışmasında önerilen yarım piksel aradeğerleme yöntemi kullanılarak elde edilen yarım piksel çözünürlükteki ikili imgeler görülmektedir. Şekil 6.3(e) ve Şekil 6.3(f)'de ise [119] ve bu tez çalışmasında önerilen çeyrek piksel aradeğerleme yöntemi kullanılarak elde edilen çeyrek piksel çözünürlükteki ikili imgeler görülmektedir.



Şekil 6.3: (a) *"mobile"* dizisinden bir çerçeve, (b) Çerçevenin 1BD sonucu, (c) [119]'da önerilen yöntem kullanılarak elde edilen yarım piksel aradeğerleme sonucu, (d) Önerilen yöntem kullanılarak elde edilen yarım piksel aradeğerleme sonucu, (e) [119]'da önerilen yöntem kullanılarak elde edilen çeyrek piksel sonucu, (f) Önerilen yöntem kullanılarak elde edilen çeyrek piksel aradeğerleme sonucu

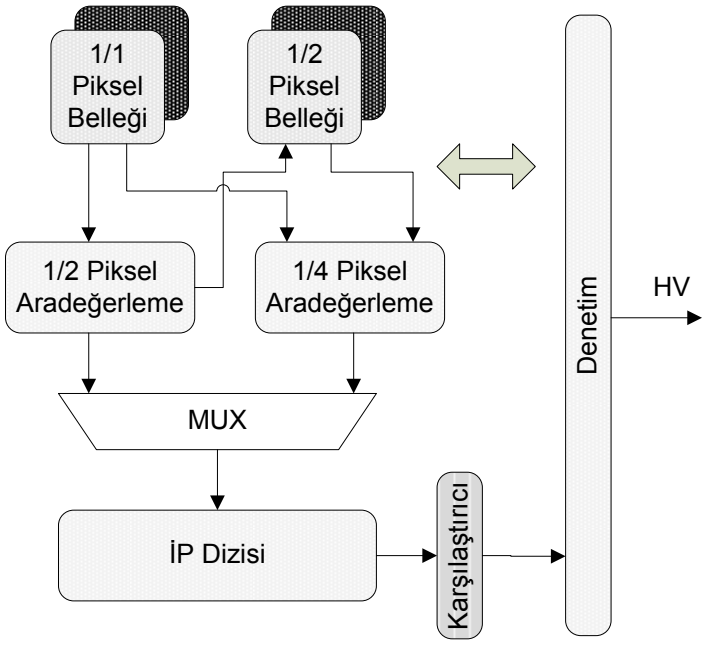
6.2 Ç1BD Temelli KHK Donanımı Mimarisi

Bu bölümde, bu tez çalışmasına özgü olan Ç1BD yöntemi ile KHK yapan yeni bir donanım mimarisi anlatılmaktadır. Mimarinin tasarımında kullanılan tamsayı HK kestirimi bloğu için Bölüm 4.2.2.2'de anlatılan KPTDD temelli mimari kullanılmıştır.

6.2.1 Ç1BD temelli yarım piksel ve çeyrek piksel HK donanımı mimarisi

Şekil 6.4'de önerilen yarım piksel ve çeyrek piksel HK donanımının mimarisi görülmektedir. 8 işlem parçasından oluşan *İP dizisi* öbeği ve *Karşılaştırıcı* öbeği yarım piksel ve çeyrek piksel HK işleminde paylaşılarak kullanılmaktadır. Sırayla ilk önce yarım piksel HK işleminde ardından çeyrek piksel HK işleminde kullanılırlar.

Yarım piksel aradeğerleme ve yarım piksel arama işlemleri ardışık düzende gerçekleştirilmektedir. Çeyrek piksel HK ile ilgili işlemler yarım piksel arama sonucuna göre şekilleneceği için bu işlem bitmeden çeyrek piksel işlemlere başlamak mümkün değildir.

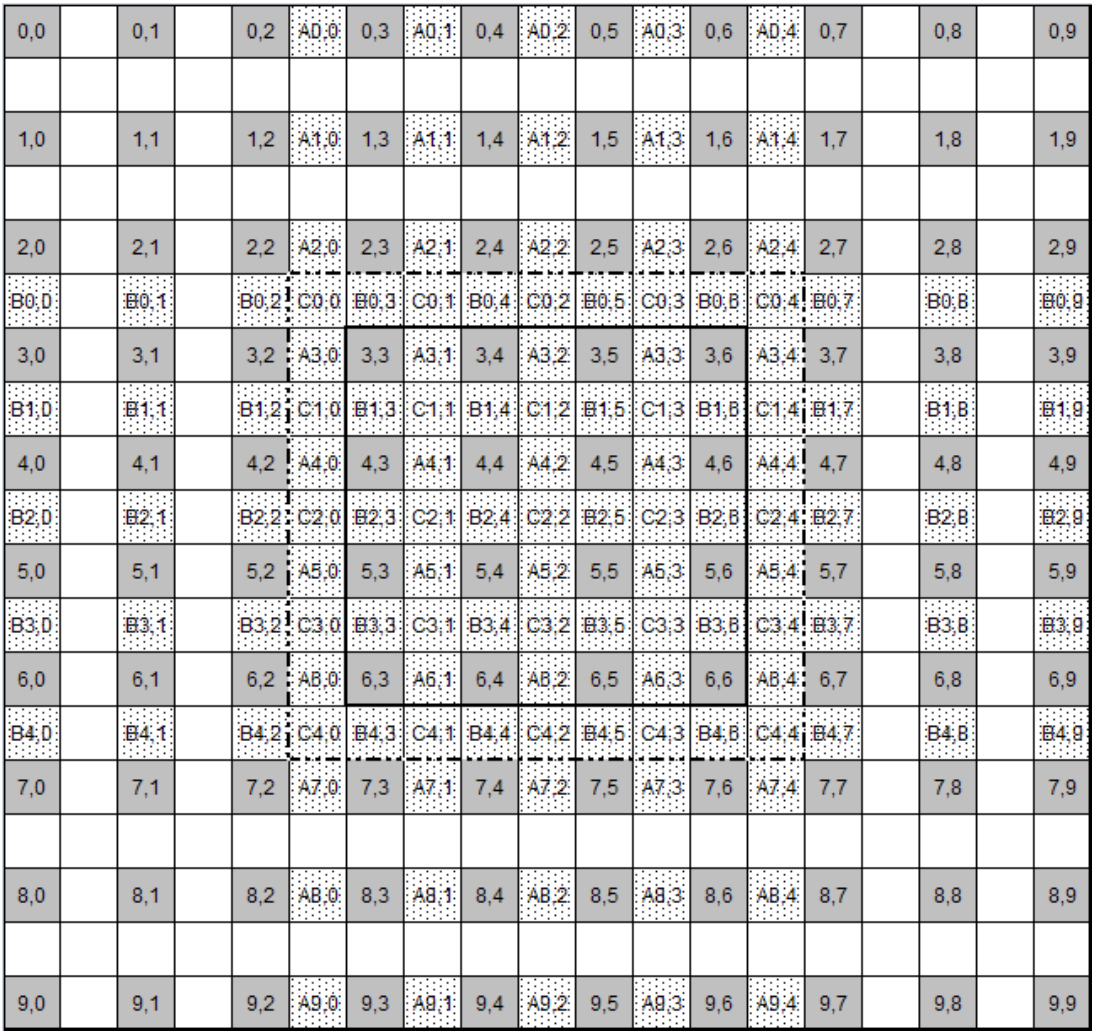


Şekil 6.4: Ç1BD temelli yarım piksel ve çeyrek piksel hareket kestirimi donanımının mimarisi

Yarım piksel HK işleminde ilk adım tamsayı piksellerden aradeğerleme işlemi ile yarım piksellerin hesaplanmasıdır. Şekil 6.5'de örnek olarak 4×4'lük bir tamsayı blok üzerinde aradeğerleme işlemi görsel olarak ifade edilmiştir. Aradeğerleme penceresinde *A* tipi, *B* tipi ve *C* tipi yarım pikseller görülmektedir. Yarım pikseller tamsayı piksellere göre bulundukları konum esas alınarak bu üç tip belirlenmiştir. Aradeğerleme işlemi şekil içindeki koyu siyah çerçeve ile sınırlandırılmış gri dolgulu 4×4 boyutlu tamsayı pikseller üzerinde gerçekleştirilmektedir. 5 adet *A* tipi yarım pikseli hesaplayabilmek için şekilde de görüldüğü gibi 10 adet tamsayı piksele ihtiyaç vardır. Burada görülen bir satırdaki 5 adet *A* tipi piksel 5 bitlik bir satır vektörü

ile aynı şekilde aradeğerleme işleminde kullanılacak 10 adet tamsayı pikselde 10 bitlik bir satır vektörü ile ifade edilmektedir. [120]'de yarım piksel aradeğerleme işlemi için 5 adet toplama bir adet çıkartma ve 2 adet kaydırma işlemi gerçekleştirilmektedir, [121]'de ise bu işlem daha yüksek performansta paralelleştirilerek gerçekleştirilmekte ve bu işlem için ise 7 adet toplama, bir adet çıkartma ve 6 adet kaydırma işlemi yapılmaktadır.

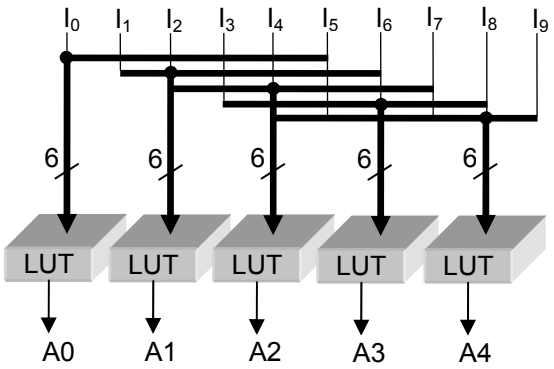
Bu tez çalışması kapsamında önerilen yarım piksel aradeğerleme donanımı ise bir piksel için aradeğerleme işlemini sadece 6 girişli bir LUT kullanarak tamamen paralel gerçekleştirmektedir. Dolayısıyla literatürdeki 8bit/piksel temelli benzer donanım mimarileri ile karşılaştırıldığında son derece basit bir yapıya sahiptir.



Şekil 6.5: 4×4'lük bir tamsayı piksel bloğu için gerekli yarım piksel aradeğerleme penceresi [121].

Bu tez çalışmasında önerilen KHK donanımı mimarisi 22×22 boyutlu bir tamsayı piksel arama penceresi ve 16×16 güncel blok belleği kullanmaktadır ancak şekli basitleştirmek amacıyla bu bellekler gösterilmemiştir. Şekilde görülen yarım piksel belleği; *A* tipi yarım pikselleri saklamak amacıyla 18×17 bit, *B* tipi yarım pikselleri saklamak amacıyla 17×18 bit ve *C* tipi yarım pikselleri saklamak amacıyla 17×17 bit boyutunda üç farklı bellek içermektedir.

Önerilen yarım piksel aradeğerleme donanımı mimarisinin 4×4 blok için tasarlanan hali Şekil 6.6'da görülmektedir. 4×4'lük bir tamsayı blok için aradeğerleme donanımında beş adet LUT kullanılmaktadır. 5 bitlik bir *A* tipi yarım piksel vektörünü hesaplayabilmek için tamsayı piksel belleğinden 10 bitlik bir okuma yapılması gerekmektedir. Okunan 10 bitlik satır vektörü aradeğerleme donanımındaki LUT öbeklerinin girişlerine uygun şekilde yönlendirilmektedir.



Şekil 6.6: Önerilen aradeğerleme donanımının 4×4 boyutlu tamsayı piksel penceresi için düzenlenmiş hali.

Her bir LUT'nin 6 bitlik bir girişi, 1 bitlik bir çıkışı vardır ve yarım piksel aradeğerleme işlemi yapar. HK işlemi 16×16'lık bir tamsayı blok için yapılmak istenirse 22×22'lik bir aradeğerleme penceresine gerek vardır ve aradeğerleme donanımında 17 adet LUT kullanılması gerekmektedir. Bu nedenle önerilen, ikili yarım piksel aradeğerleme donanımı mimarisinde bir saat darbesinde 17 adet *A* tipi yarım piksel satır vektörünü hesaplayabilmek için 17 adet LUT'nin girişine 22 bitlik tamsayı piksel satır vektörü uygulanır dolayısıyla bir yarım piksel yerine 17 yarım piksel tek bir saat darbesinde hesaplanmış olur. Her bir LUT bir saat darbesinde bir tane yarım piksel üretmektedir.

Tasarlanan mimaride tamsayı pikseller bellekten satır satır okunmaktadır. *B* tipi yarım pikseller tamsayı piksellerin dikey komşuluğunda bulunmaktadırlar. Bu tip yarım piksellerin hesaplanabilmesi için okunan tamsayı pikseller kaydırmalı saklayıcılar üzerinden seri bir şekilde kaydırıldıktan sonra paralel bir şekilde bu saklayıcılardan okunarak aradeğerleme donanımının girişlerine uygulanmaktadırlar. Aynı satırda bulunan pikseller farklı kaydırmalı saklayıcılara gönderilmektedirler.

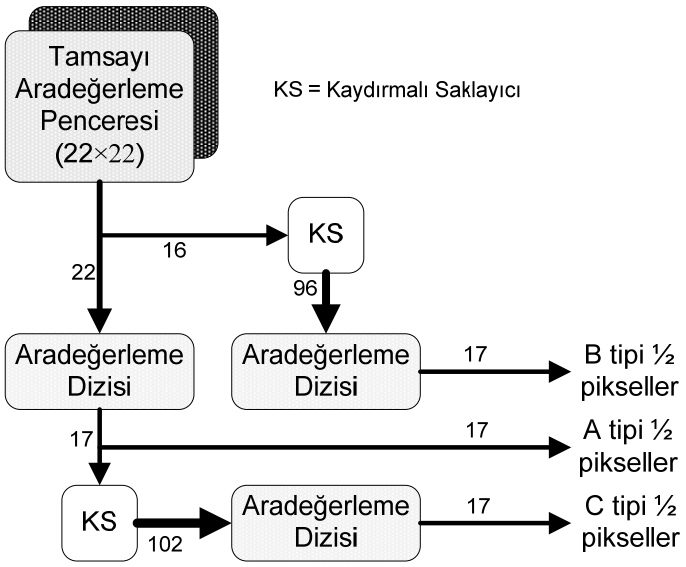
Dolayısıyla 16×16 boyutunda bir tamsayı bloğa ait *B* tipi yarım piksellerin hesaplanması için 16 adet 6 bitlik kaydırmalı saklayıcı kullanılması gerekmektedir. Bu nedenle *B* yarım piksel aradeğerleme dizisine giden veri hattının genişliği $16 \times 6 = 96$ bittir.

C tipi yarım piksellerin hesaplanması için yarım iki tip yarım pikselden herhangi birisi kullanılabilir ancak, *A* tipi yarım piksellerin hesaplanması daha kısa sürdüğü için önerilen KHK mimarisinde *C* tipi yarım pikseller *A* tipi yarım pikseller kullanılarak hesaplanmaktadır.

B tipi yarım piksellerin hesaplanmasında olduğu gibi, *C* tipi yarım piksellerin hesaplanmasında da paralel çıkışlı seri kaydırmalı saklayıcılar kullanılmaktadır. Ancak burada kaydırmalı saklayıcılara tam sayı pikseller değil, *A* tipi yarım pikseller seri olarak girmekte ve paralel olarak çıktıktan sonra aradeğerleme veri yoluna gitmektedirler.

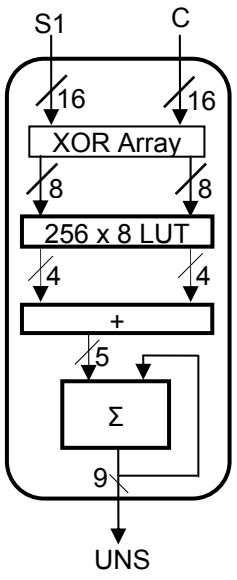
Önerilen yarım piksel aradeğerleme donanımı mimarisi Şekil 6.7'de görülmektedir. Donanım önce *A* tipi yarım pikselleri hesaplamaktadır. Kaydırma işlemi nedeniyle *B* ve *C* tipi yarım piksellerin hesaplanması gecikmeli olarak gerçekleşmektedir. Tüm yarım piksel aradeğerleme işlemi toplam 22 saat darbesi sürmektedir ve bu süre aradeğerleme belleğinin genişliğine bağlıdır.

Önerilen KHK donanımı mimarisi, sadece 8 farklı yarım piksel arama noktası olduğu için bu işi çok kısa bir sürede paralel olarak gerçekleştirebilmektedir.



Şekil 6.7: Ç1BD temelli yarım piksel aradeğerleme donanımı mimarisi

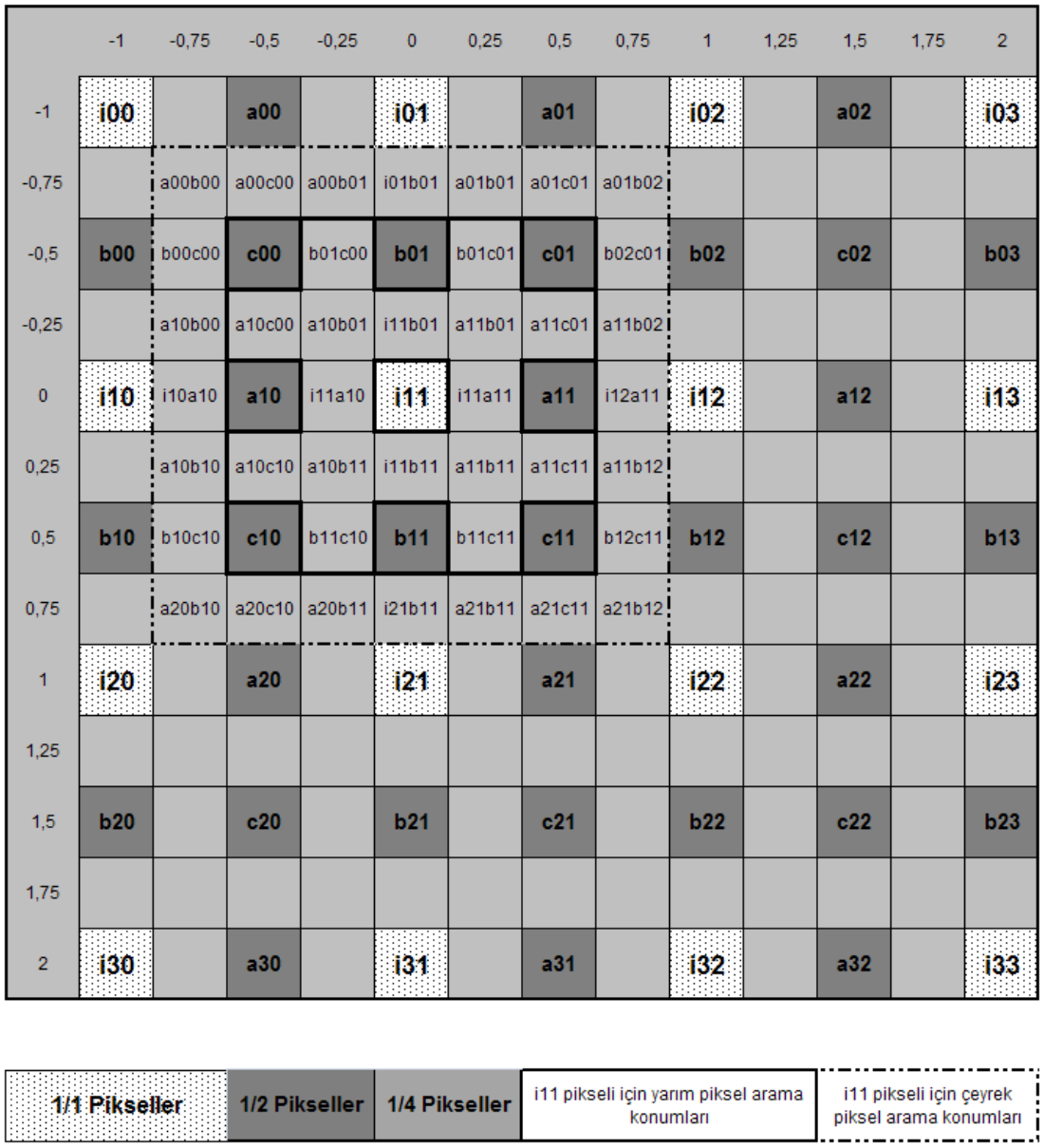
KHK işleminde kullanılan İP mimarisi Şekil 6.8'de görülmektedir.



Şekil 6.8: Ç1BD temelli KHK işleminde kullanılan İP mimarisi

İlk yarım piksel satır vektörü, yarım piksel aradeğerleme donanımının çıkışında görülür görülmez yarım piksel arama işlemi başlatılır. Bu nedenle en son yarım piksel satır vektörünün çıkması ile birlikte son UNS değeri 24. saat darbesinin çıkan kenarında hesaplanır. Yarım piksel uyumlama işlemi 21. saat darbesinin çıkan kenarı ile başlar çünkü ilk yarım piksel arama konumunun UNS değeri bu noktada hazır hale gelmiştir. 27. saat darbesinin çıkan kenarında yarım piksel arama vektörleri çıkışlara yazılır.

Çeyrek piksel aradeğerleme işlemi 47. saat darbesinin çıkan kenarında tamamlanır ve sonunda 49. saat darbesinin çıkan kenarında çeyrek piksel hareket vektörleri karşılaştırıcı öbeğinin çıkışına yazılır. Ç1BD temelli çeyrek piksel HK işleminde yarım piksel HK işleminin sonucuna bağlı olarak 9 farklı çeyrek piksel aradeğerleme konumu vardır. Şekil 6.9'da 2×2 boyutlu bir tamsayı bloğu için çeyrek piksel aradeğerleme senaryosu gösterilmiştir. Çeyrek piksel aradeğerleme işleminde hem tamsayı hem de yarım piksellerin kullanılması gerekmektedir. Şekil 6.9'da koyu çizgi ile sınırlandırılmış bölge i11 tamsayı pikseli etrafındaki 9 olası aradeğerleme konumunun tümünü kapsamaktadır. Kesikli çizgi ile sınırlandırılmış pencere 9 farklı yarım piksel hareket vektörü konumu için 9 farklı çeyrek piksel arama gölgesinin tümünü kapsamaktadır. Dolayısıyla şekilde 81 farklı çeyrek piksel arama konumu görünmektedir.



Şekil 6.9: 2×2 boyutlu tamsayı piksel bloğu için kullanılması gereken çeyrek piksel aradeğerleme penceresi.

Tablo 6.9'da çeyrek piksel aradeğerleme yöntemi görülmektedir. Ç1BD yönteminin ikili doğası gereği çeyrek piksel aradeğerleme işlemi basit iki girişli VEYA kapısı ile gerçekleştirilmektedir. Tablodan da görüleceği gibi sekiz farklı arama konumu için (AK0-AK7) çeyrek piksellerin yarım piksel arama işleminin sonucuna göre hesaplanması gerekmektedir.

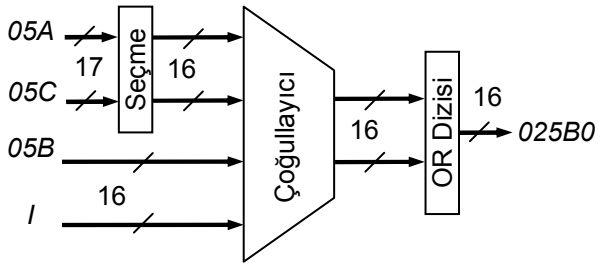
Yarım piksel arama işleminin olası 9 farklı sonucuna bağlı olarak çeyrek piksel aradeğerleme işlemine girmesi gereken değişkenler değişmektedir ve her bir değişkenin hesaplanması için 9 ayrı veri-yolu tasarlanması gerekmektedir. Örneğin yarım piksel arama sonucu elde edilen yarım piksel HV değeri (-1,1) olursa AN3 çeyrek piksel arama konumu için çeyrek piksel aradeğerleme değişkenleri a20 ve c10 olur.

Tablo 6.2: Çeyrek piksel aradeğerleme tekniği

Yarım piksel arama sonuçlarına bağlı olarak farklı çeyrek piksel hesaplama kombinasyonları									
1/4 piksel arama konumları	**i11(x,y) tamsayı pikseli için 1/2 piksel hareket vektörü konumları**								
	(-1,-1)	**(0,-1)**	**(1,-1)**	**(-1,0)**	**(0,0)**	**(1,0)**	**(-1,1)**	**(0,1)**	**(1,1)**
AN0	b10,c10	b11,c10	b11,c11	i10,a10	i11,a10	i11,a11	b00,c00	b01,c00	b01,c01
AN1	b11,c10	b11,c11	b12,c11	i11,a10	i11,a11	i12,a11	b01,c00	b01,c01	b02,c01
AN2	a10,c10	i11,b11	a11,c11	a10,c00	i11,b01	a11,c01	a00,c00	i01,b01	a01,c01
AN3	a20,c10	i21,b11	a21,c11	a10,c10	i11,b11	a11,c11	a10,c00	i11,b01	a11,c01
AN4	a10,b10	a10,b11	a11,b11	a10,b00	a10,b01	a11,b01	a00,b00	a00,b01	a01,b01
AN5	a10,b11	a11,b11	a11,b12	a11,b00	a11,b01	a11,b02	a00,b01	a01,b01	a01,b02
AN6	a20,b10	a21,b11	a21,b11	a10,b10	a10,b11	a11,b11	a10,b00	a10,b01	a11,b01
AN7	a20,b11	a21,b11	a21,b12	a10,b11	a11,b11	a11,b12	a10,b00	a11,b01	a11,b02

AN4	AN2	AN5
AN0	YP	AN1
AN6	AN3	AN7

Önerilen donanım mimarisinde, 8 farklı konumdaki her bir çeyrek piksel için farklı bir veri yolu kullanılarak hesaplama yapılır. Bu nedenle 8 farklı aradeğerleme veri-yolu tasarlanmıştır. Çeyrek piksel arama noktalarından AN2 konumundaki çeyrek piksel için kullanılan veri-yolu mimarisi Şekil 6.10'da görülmektedir. Diğer çeyrek piksellerin hesaplanması için kullanılan veri-yolları küçük farklılıklar dışında aynı denilebilir. Tablo 6.2'deki veriler kullanılarak diğer veri-yolu mimarileri de kolaylıkla tasarlanabilir.



Şekil 6.10: AN2 çeyrek piksel arama konumu için kullanılan çeyrek piksel aradeğerleme veri-yolu

Şekil 6.10'da görülen *Seçme* öbeği hareket vektörünün düşey bileşeni doğrultusunda girişindeki *A* ve *C* tipi piksel vektörlerinin uygun bir bölümünü seçmek için kullanılır.

A ve *C* tipi yarım piksel vektörlerinin seçilmiş bölümleri ile *B* tipi yarım piksel vektörü ve tamsayı piksel vektörü bir tane çoğullayıcı öbeğine girmektedir. *Çoğullayıcı* öbeği ise yarım piksel hareket vektörünün yatay bileşeni doğrultusunda dört farklı girişinden ikisini çeyrek piksel aradeğerleme işlemi için çıkışına göndermektedir. Eğer yarım piksel hareket vektörünün yatay bileşeni sıfır ise bu iki çıkışa *B* ve *I* vektörleri, diğer durumlarda ise *A* ve *C* vektörleri gidecektir.

6.3 Deneysel Sonuçlar

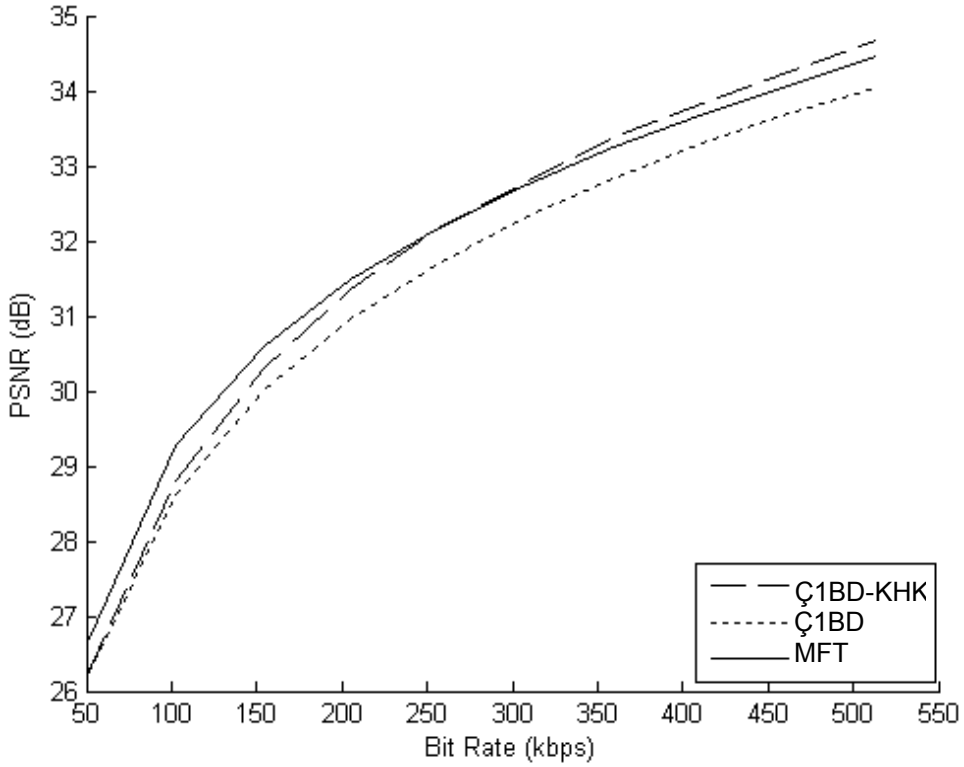
Önerilen tamamen ikili KHK yönteminin kapalı çevrim ve açık çevrim testleri gerçekleştirilmiştir. Açık çevrim test düzeninde güncel video çerçevesi bir önceki çerçeve kullanılarak yeniden oluşturulmuştur ve asıl imge ile yeniden oluşturulan imge arasındaki bozulma, PSNR ölçütü kullanılarak ölçülmüştür.

Tablo 6.3'de diğer 1BD temelli THK yöntemleri ile bu çalışmada önerilen 1BD temelli KHK yöntemi için elde edilen açık çevrim test sonuçları görülmektedir. Sonuçlardan anlaşılacağı üzere önerilen, Ç1BD temelli KHK yöntemi tamamen ikili olarak gerçekleştirilen aradeğerleme işlemleri sayesinde elde edilen piksel altı çözünürlüğü nedeniyle Ç1BD temelli THK işleminin başarımını ortalama 0.3 dB arttırmıştır. Önerilen tamamen ikili KHK yöntemi yüksek detaylı bir örüntüye sahip olan *"mobile"* dizisinde, MFT ölçütünden daha iyi bir başarım göstermiştir.

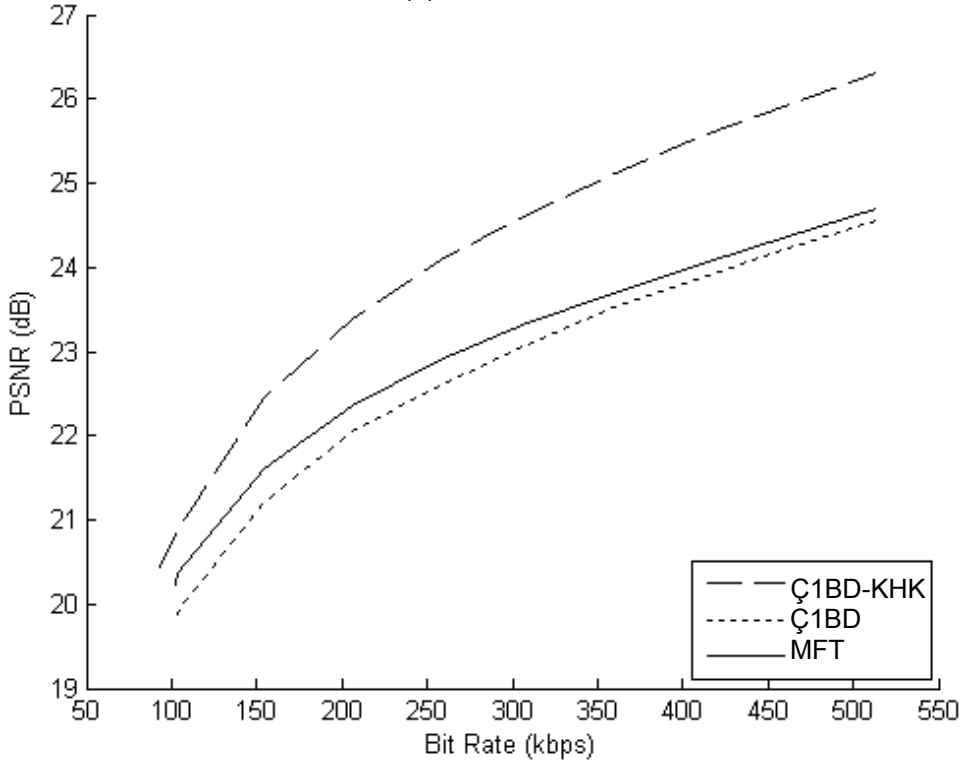
Tablo 6.3: Çeşitli HK yöntemleri ile 16×16 blok boyutu ve $[-16,+16]$ arama aralığında tam arama yapılarak yeniden oluşturulan video dizilerinin ortalama PSNR değerleri

Yöntem	Video sizisi adı, çerçeve boyutu, dizi uzunluğu					
	"football" 352×240 125	"foreman" 352×288 300	"tennis" 352×240 150	Garden 352×240 115	"mobile" 352×240 300	"coastguard" 352×288 300
MFT	22.88	32.09	29.45	23.79	23.94	30.48
MFT_025	23.79	34.09	30.43	25.48	26.52	32.03
1BD[48]	21.83	30.32	28.11	23.31	23.61	29.83
Ç1BD[51]	21.81	30.38	28.18	23.26	23.63	29.88
Ç1BD-KHK	**22.03**	**30.62**	**28.39**	**23.67**	**24.31**	**29.99**

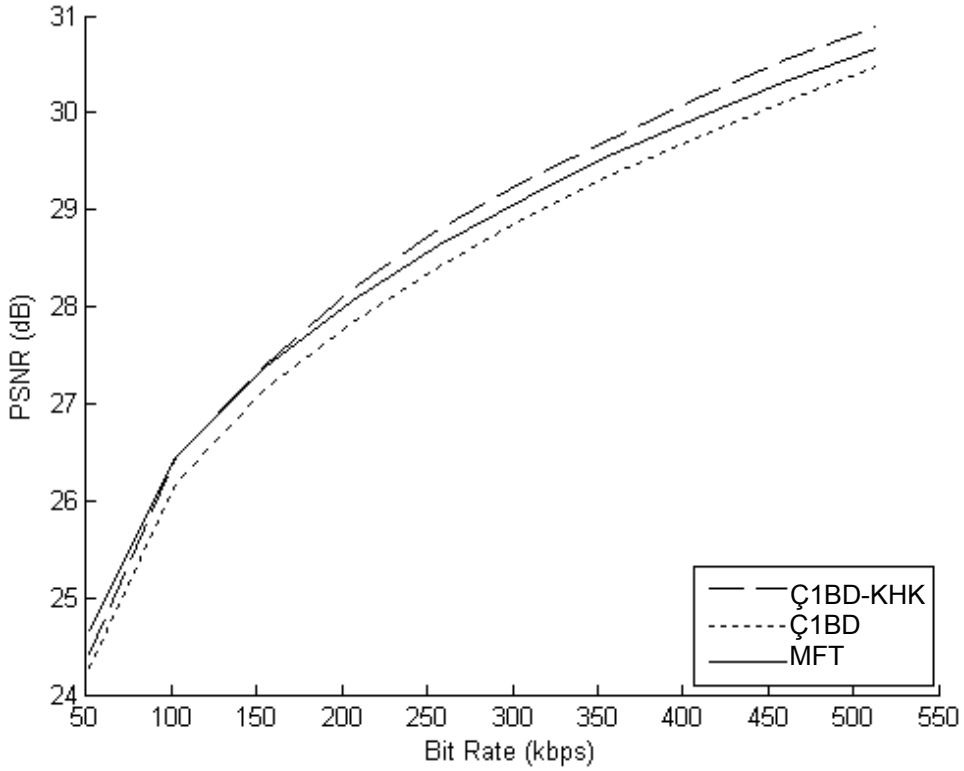
Kapalı çevrim testleri JM v12.4 kaynak yazılımı kullanılarak H.264/AVC kodlayıcı üzerinde gerçekleştirilmiştir [122]. Şekil 6.11'de *"foreman", "coastguard", "mobile"* ve *"tennis"* dizileri için gerçekleştirilen kapalı çevrim test sonuçları görülmektedir. Sonuçlardan da görüleceği üzere, önerilen Ç1BD temelli KHK yöntemi H.264 video kodlamada Ç1BD temelli THK işleminin başarımını gözle görülür bir şekilde artırmıştır. Bununla birlikte, önerilen yöntem orta ve yüksek bit oranlarında MFT temelli THK işleminden daha iyi bir başarıma sahiptir. Bu noktada H.264'ün artık kodlama başarımının 1BD yönteminden kaynaklanan ufak kestirim hatalarını dengeleyebildiğini hatırlatmakta fayda vardır.



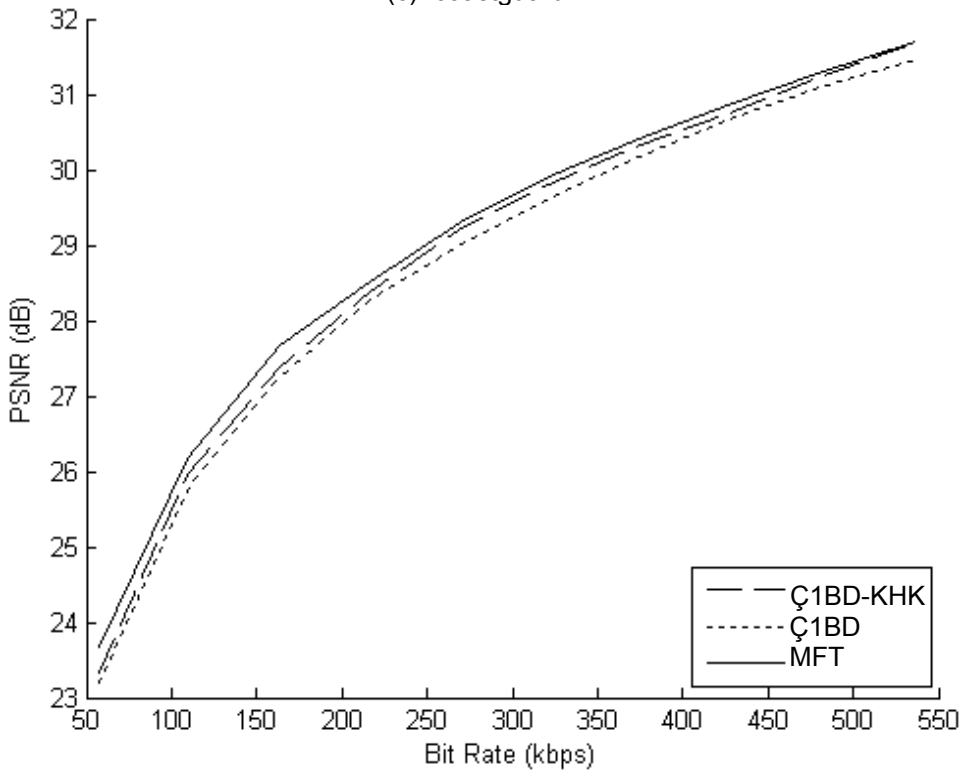
(a) “foreman”



(b) “mobile”



(c) “coastguard”



(d) “tennis”

Şekil 6.11: Önerilen yöntemin H.264/AVC kodlayıcısının referans yazılımı [122] ile test edilmesi ile elde edilen sonuçları a) “foreman”, b) “mobile”, c) “coastguard”, d)”tennis” dizileri için elde edilen sonuçlar

Önerilen Ç1BD temelli KHK yöntemi donanımı mimarisi Verilog HDL kullanılarak gerçeklenmiştir ve Mentor Graphics ModelSim benzetim aracı kullanılarak da doğrulanmıştır. Önerilen KHK donanımı mimarisi sahip olduğu tamamen ikili yapısı nedeniyle son derece düşük bir donanım karmaşıklığına sahiptir. [123]'de önerilen çalışmada bir MB için KHK işlemi 1472 saat darbesi sürmekte iken bu tez çalışmasında önerilen mimaride aynı işlem sadece 49 saat darbesinde tamamlanmaktadır. THK işleminin 1039 saat darbesi sürdüğü düşünülürse bu gayet küçük bir zaman dilimidir. Geliştirilen 1BD temelli donanım mimarisi 8bit/pixel temelli bir mimari ile karşılaştırıldığında son derece sade ve hızlı bir donanım mimarisi tasarlanmış olmaktadır.

KHK donanımı Synopsys Synplify Pro sentezleme aracı ile XC2VP30 FPGA yongası hedeflenerek tek başına 192 MHz'de çalışabilecek şekilde sentezlenmiştir. THK donanımı ise aynı araç ve hedef yonga ile 187 MHz'de çalışabilecek şekilde sentezlenmiştir. Tasarlanan KHK donanımı XC2VP30 FPGA yongasında toplam LUT sayısının %4'ünü, toplam BlockRAM sayısının da yaklaşık %6'sını kullanmıştır.

Olası ardışık düzen çalışma şeklinde tamsayı ve kesirli HK donanımı, 720p çözünürlükte 30 $çerçeve/saniye$ hızında gerçek zamanlı çalışma performansına sahiptir. [124]'de, 16CIF çözünürlükte yarım piksel doğrulukta HK için benzer bir performans elde edilebilmiştir. 720p çözünürlüğünde bu performansı sağlayabilmek için gerekli çalışma frekansı [(720×1280) / (16×16)] × (1039+49) × 30 $çerçeve/saniye$ = 117,504 MHz'dir. Oysa önerilen donanım mimarisinin çalışması 187 MHz için doğrulanmıştır.

Elde edilen sonuçlara bakıldığında önerilen tamamen ikili tamsayı piksel ikili HK yöntemi, kestirim başarımını gözle görülür şekilde artırmaktadır.

7. SONUÇLAR

Bu tez çalışması kapsamında, düşük bit derinliği gösterimi temelli yöntemlerin hareket kestirimi problemine getirdiği kolaylıklar donanım mimarisi boyutu da göz önünde bulundurularak incelenmiştir. Literatürde hareket vektörü temelli doğrusal diziler ve kaynak piksel temelli doğrusal diziler olmak üzere iki farklı mimarinin benimsendiği görülmektedir. Bu tez çalışmasında, her iki mimari kullanılarak, Ç1BD ve K-1BD temelli HK yöntemleri için donanım gerçeklemeleri yapılmıştır.

Bu tez çalışması kapsamında tasarlanan mimarilerin doğrulanması için, geliştirilen hareket kestirimi algoritmalarının bilgisayarda koşturulması ile elde edilen hareket vektörleri, benzetim sonucunda donanımın ürettiği hareket vektörleri ile karşılaştırılmıştır. Hareket kestirimi algoritmaları C dili ile kodlanmıştır. Donanımın benzetimi için Mentor Graphichs ModelSim yazılımı kullanılmıştır ve donanım tasarımında Verilog donanım tanımlama dili kullanılmıştır.

KPTDD mimarisinin düşük bit derinliğinde gösterime uygulanması ilk kez bu tez kapsamında ele alınmış olup, bu yaklaşımı kullanarak elde edilen özgün veri akışı yapısı sayesinde bellekten veri okuma miktarının düşeceği ve bunun sonucunda güç tüketiminin önemli ölçüde azaltılacağı öngörülmüş ve bu öngörülerin doğruluğu, elde edilen deneysel sonuçlar ile gösterilmiştir. Ayrıca sentez raporları göstermiştir ki bu tez çalışması kapsamında ilk kez düşük bit gösterimi temelli HK yöntemi için kullanılan KPTDD mimarisi daha az donanım karmaşıklığına sahip olduğu için daha az mantıksal kapı kullanılarak gerçeklenebilmektedir.

Yapılan tasarımlar sonucunda Ç1BD ve K-1BD temelli HK yöntemlerini gerçekleştiren donanımlarının çalışabildikleri en yüksek frekansın birbirlerine yakın olduğu gözlenmiştir. İleriye dönük olarak, her iki yöntemin HK başarımları bilindiğinden, uygulamaya bağlı olarak her iki yöntemi de kullanabilen melez bir tasarım yapılması, dolayısıyla değişen performans gereksinimlerinin karşılanması mümkün gözükmektedir.

Bu tez çalışmasında, Gray kodlama temelli yeni bir HK yöntemi önerilmiştir. Video çerçevelerinin Gray kodlanması sonrasında bit kesme tekniği kullanılarak çerçevelerin bit derinliği azaltılmıştır. Ardından, BDU tekniği uygulanarak HK işlemi gerçekleştirilmiştir. Yöntem, temelde 1BD kullanmamasına karşın, 1BD temelli HK yöntemi için geliştirilmiş bir donanım mimarisi temel alınarak düşük karmaşıklığa sahip yüksek performanslı yeni bir donanım mimarisi önerilmiştir.

Bu çalışmada son olarak özgün, tamamen ikili bir KHK yöntemi ve donanım mimarisi önerilmiştir. Elde edilen deneysel sonuçlar göstermiştir ki önerilen yöntem 1BD temelli THK yöntemlerinden daha iyi HK başarımına sahiptir. Ayrıca önerilen KHK donanımı mimarisi, tam sayı hareket kestirimi donanımı ile bir arada kullanıldığında, THK donanımına önemli bir ek yük getirmediğinden olası bir tümleşik HK donanımının performansına olumsuz bir etkisi olmayacaktır.

Bu tez çalışması kapsamında geliştirilen bütün yöntem ve mimariler, daha az veri kullanılarak ve aritmetik yerine mantıksal işlemlere ağırlık verilerek tasarlandığından, pil destekli uygulamalar gibi önemli güç kısıtlamaları ve sınırlı işlem kapasitesine sahip video kamera ve mobil telefon gibi tüketici elektroniği aygıtlarında kullanılmak için son derece uygundurlar.

Önerilen yöntem ve mimarilerin, güncel video kodlama standartlarının önemli bir bileşeni olan değişken blok boyutlu HK işlemine uyarlanması gelecekte yapılabilecek olası geliştirmeler içerisindedir. Ayrıca diğer düşük bit derinliği gösterimi temelli HK yöntemlerine göre oldukça yüksek başarım sağlayan KGKBDU yönteminin piksel altı doğrulukta HK işlemine uyarlanması da gelecekte yapılabilecek çalışmalar arasındadır.

KAYNAKLAR

[1] Richardson, I., "Video Codec Design: Developing Image and Video Compression Systems," ***John Wiley & Sons***, First Edition, (2002)

[2] Information Technology-Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5 Mbit/s—Part 2: Video, ***ISO/IEC 11172-2***, (1993).

[3] Information Technology-Generic Coding of Moving Pictures and Associated Audio Information: Video, ***ISO/IEC 13818-2 and ITU-T Recommendation H.262***, (1996).

[4] Information Technology-Coding of Audio-Visual Objects-Part 2: Visual, ***ISO/IEC 14496/2***, (1999).

[5] Video Codec for Audiovisual Services at $p \times 64Kbit/s$, ***ITU-T Recommendation H.261***, (1993).

[6] Video Coding for Low Bit Rate Communication, ***ITU-T Recommendation H.263***, (1998).

[7] Joint Video Team, Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification, ***ITUT Recommendation H.264 and ISO/IEC 14496/10 AVC***, (2003).

[8] Yao Wang, Jörn Osterman, Ya-Qin Zhang, "Video Processing and Communications, ***Prentice Hall***, (2002).

[9] Chen, C., Chien, S., Huang, Y., Chen, T., Wang, T., Chen, L., "Analysis and Architecture Design of Variable Block Size Motion Estimation for H.264/AVC," ***IEEE Transactions on Circuits and Systems I: Regular Papers***, 3, 578 – 593, (2006).

[10] Komarek, T.; Pirsch, P., "Array architectures for block matching algorithms," ***IEEE Transactions Circuits and Systems***, 10, 1301-1308, (1989).

[11] Yeh, Y.H., Lee, C.Y., "Cost-effective VLSI architectures and buffer size optimization for full-search block matching algorithms," ***IEEE Transactions on Very Large Scale Integration (VLSI) Systems***, 3, 345-358, (1999).

[12] Murachi, Y., Mizuno, K., Miyakoshi, J., Hamamoto, M., Iinuma, T., Ishihara, T., Yin, F., Lee, J.; Kamino, T.; Kawaguchi, H.; Yoshimoto, M., "A sub 100 mW H.264/AVC MP@L4.1 integer-pel motion estimation processor VLSI for MBAFF encoding," ***IEEE International Symposium on Circuits and Systems***, 848 – 851, (2008).

[13] Parhi, K., "VLSI Digital Signal Processing Systems: Design And Implementation," First Edition, ***John Wiley & Sons***, (1999).

[14] Huang, Y.W., Chen, C.Y., Tsai, C.H., Shen, C.F., Chien, L.G., “Survey on Block Matching Motion Estimation Algorithms and Arhitectures with New Results,” ***Journal of VLSI Signal Processing,*** 42, 297–320, (2006).

[15] Jain, J., Jain, A., “Displacement Measurement and its Application in Internal Image Coding,” ***IEEE Trans. Commun.***, 12, 1799–1808, (1981).

[16] Koga, T., Linuma, K., Hirano, A., Lijima, Y., Ishiguro, T., “Motion compensated interframe coding for video conferencing***,” In Proc. Nat. Telecommun. Conf.***, C9.6.1–C9.6.5, (1981).

[17] Srinivasan, R., Rao, K.R., “Predictive Coding based on Efficient Motion Estimation,” ***IEEE Trans. Commun.***, 8, 888–896, (1985).

[18] Kappagantula, S., Rao, K.R., “Motion Compensated Inter frame Image Prediction,” ***IEEE Trans. Commun.***, 9, 1011–1015,(1985).

[19] Ghanbari, M., “The Cross Search Algorithm for Motion Estimation,” ***IEEE Trans. Commun.***, 7, 950–953, (1990).

[20] Chen, L.G., Chen, W.T., Jehng, Y.S., Chiueh, T.D., “An Efficient Parallel Motion Estimation Algorithm for Digital Image Processing,” ***IEEE Trans. Circuits Syst. Video Technol.***, 4, 378–385, (1991).

[21] Chen, M.J., Chen, L.G., Chiueh, T.D., “One-dimensional full Search Motion Estimation Algorithm for Video Coding,” ***IEEE Trans. Circuits Syst. Video Technol.***, 5, 504–509, (1994).

[22] Li, R., Zeng, B., Liou, M.L., “A New Three-step Search Algorithm for BlockMotion Estimation,” ***IEEE Trans. Circuits Syst. Video Technol.***, 4, 438-442, (1994).

[23] Po, L.M., Ma, W.C., “A Novel Four-step Search Algorithm for Fast Block Motion Estimation,” ***IEEE Trans. Circuits Syst. Video Technol.***, 3, 313–317, (1996).

[24] Tham, J.Y., Ranganath, S., Ranganath, M., Kassim, A.A., “A Novel Unrestricted Center-biased Diamond Search Algorithm forBlock Motion Estimation,” ***IEEE Trans.Circuits Syst. Video Technol.***, 4, 369–377, (1998).

[25] Zhu, S., Ma, K.K., “A New Diamond Search Algorithm for Fast Block-matching Motion Estimation,” ***IEEE Trans. Image Processing***, 2,. 287–290, (2000).

[26] Tourapis, A.M., Au, O.C., Liu, M.L., “Highly Efficient Predictive Zonal Algorithms for Fast Block-matching Motion Estimation,” ***IEEE Trans. Circuits Syst. Video Technol.***, 10, 934–947, (2002).

[27] Christopoulos, V., Cornelis, J., “A Center-biased Adaptive Search Algorithm for Block Motion Estimation,” ***IEEE Trans. Circuits Syst. Video Technol.***, 3, 423–426, (2000).

[28] Cheung, C.H., Po, L.M., “A Novel Cross Diamond Search Algorithm for Fast Block Motion Estimation,” ***IEEE Trans. Circuits Syst. Video Technol***., 12, 1168– 1177, (2002).

[29] Huang, Y.W., Ma, S.Y., Shen, C.F., Chen, L.G., “Predictive Line Search: An Efficient Motion Estimation Algorithm for mpeg-4 Encoding Systems on Multimedia Processors,” ***IEEE Trans. Circuits and Syst. Video Technol.***, 1, 111–117, (2003).

[30] Chen, O.T.C., “Motion Estimation using a One-Dimensional Gradient Descent Search,” ***IEEE Trans. Circuits Syst. Video Technol.***, 4, 608–616, (2000).

[31] C. Zhu, X. Lin, and L. P. Chau, “Hexagon-based search pattern for fast block motion estimation,” ***IEEE Trans. Circuits Syst. Video Technol.*** 15, 349–355 , (2002).

[32] C. Zhu, X. Lin, L. P. Chau, and L. M. Po, “Enhanced hexagonal search for fast block motion estimation,” ***IEEE Trans. Circuits Syst. Video Technol.***, 10, 1210–1214, (2004).

[33] So, H.; Kim, J.; Cho, W.-K.; Kim, Y.-S.; “Fast motion estimation using modified diamond search patterns,” ***Electronics Letters***, 2, 62 – 63, (2005).

[34] Tsai, T.-H., Pan, Y.-N., “A Novel 3-D Predict Hexagon Search Algorithm for Fast Block Motion Estimation on H.264 Video Coding,” ***IEEE Transactions on Circuits and Systems for Video Technology,*** 12, 1542-1549, (2006).

[35] Yu, C.-S., Tai, S.-C., **“**Adaptive Double-Layered Initial Search Pattern for Fast Motion Estimation,” ***IEEE Transactions on Multimedia***, 6, 1109-1116, (2006).

[36] Po, L.-M., Ting, C.-W., Wong, K.-M., Ng, K.-H., “Novel Point-Oriented Inner Searches for Fast Block Motion Estimation,” ***IEEE Transactions on Multimedia***, 1, 9-15, (2007).

[37] Kim, K.B., Jeon, Y., Hong, M.-C., “Variable step search fast motion estimation for H.264/AVC video coder,” ***IEEE Transactions on Consumer Electronics***, 3, 1281-1286, (2008).

[38] Gonzalez-Diaz, I., Diaz-de-Maria, F., “Adaptive Multipattern Fast Block-Matching Algorithm Based on Motion Classification Techniques,” IEEE Transactions on ***Circuits and Systems for Video Technology,*** 10, 1369-1382, (2008).

[39] Bierling, M., “Displacement Estimation by Hierarchical Block Matching,” ***in Proc. of SPIE Visual Commun. Image Processing (VCIP’88)***, 942–951, (1998).

[40] Zaccarin, A., Liu, B., “Fast Algorithms for Block Motion Estimation,” ***in Proc. of IEEE International Conference Acoustic, Speech, and Signal Processing (ICASSP’92)***, San Francisco, CA, USA, 3, 449–452, 23-26 March, (1992).

[41] Liu, B., Zaccarin, A., “New Fast Algorithms for the Estimation of Block Motion Vectors,” ***IEEE Trans. Circuits Syst. Video Technol***., 2, 148–157, (1993).

[42] Wang, Y., Wang, Y., Kuroda, H., “A Globally Adaptive Pixel decimation Algorithm for Block-motion Estimation,” ***IEEE Trans. Circuits Syst. Video Technol***., 6, 1006–1011, (2000).

[43] Gharavi, H., Mills, M., “Block Matching Motion Estimation Algorithms - New Results,” ***IEEE Trans. Circuits Syst***., 5, 649–651, (1990).

[44] Chen, M.J., Chen, L.G., Chiueh, T.D., Lee, Y.P., “A New Block-matching Criterion for Motion Estimation and its Implementation,” ***IEEE Trans. Circuits Syst. Video Technol***., 3, 231–236, (1995).

[45] Kim, J.S., Park, R.H., “A Fast Feature-based Block Matching Algorithm using Integral Projections,” ***IEEE J. Select. Areas Commun***., 5, 968–979, (1992).

[46] Sauer, K., Schwartz, B., “Efficient Block Motion Estimation using Integral Projections,” ***IEEE Trans. Circuits Syst. Video Technol***., 5, 513–518, (1996).

[47] Feng, K.-T. Lo, Mehrpour, H., Karbowiak, A. E., “Adaptive Block Matching Motion Estimation Algorithm using Bit-Plane Matching,” ***in Proc. International Conference on Image Processing***, Washington, USA, 496–499, 23-26 October, (1995).

[48] Natarajan, B., Bhaskaran, V., “Low-complexity Block-based Motion Estimation via one-bit Transforms,” ***IEEE Trans. Circuits Syst. Video Technol***., 4, 702–706, (1997).

[49] Luo, J.H., Wang, C.N., Chiang, T., “A Novel All-binary Motion Estimation (ABME) with Optimized Hardware Architectures,” ***IEEE Trans. Circuits Syst. Video Technol***., 8, 700–712, (2002).

[50] Ertürk, A., Ertürk, S., “Two-Bit Transform for Binary Block Motion Estimation,” ***IEEE Trans. Circuit Syst. Video Technol***., 7: 938-946, (2005).

[51] Erturk, S., "Multiplication-Free One-Bit Transform for Low-Complexity Block-Based Motion Estimation,” ***Signal Processing Letters, IEEE*** ,.2, 109-112, (2007).

[52] Urhan, O.; Erturk, S., "Constrained One-Bit Transform for Low Complexity Block Motion Estimation,” ***Circuits and Systems for Video Technology, IEEE Transactions on*** , 4, 478-482, (2007).

[53] Urhan, O.,"Constrained One-Bit Transform Based Motion Estimation using Predictive Hexagonal Pattern,” ***Journal of Electronic Imaging***, 3, (2007).

[54] He, Z.L., Tsui, C.Y., Chan, K.K., Liou, M.L., “Low-power VLSI Design for Motion Estimation using Adaptive Pixel truncation,” ***IEEE Trans. Circuits Syst. Video Technol.***, 5, 669–678, (2000).

[55] Bhaskaran, V., Konstantinides, K., “Image and Video Compression Standards: Algorithms and Architectures,” ***Kluwer Academic Publishers***, (1997).

[56] Hsieh, C.H., Lu, P.C., Shyn, J.S., Lu, E.H., “Motion Estimation Algorithm using Inter block Correlation,” ***IEE Electron. Lett.***, 5, 276–277, (1990).

[57] Zafar, S., Zhang, Y.Q., Baras, J.S., “Predictive Block Matching Motion Estimation for TV Coding—Part I: Inter-block Prediction,” ***IEEE Trans. Broadcast.***, 3,.97–101, (1991).

[58] Zhang Y.Q., Zafar, S., “Predictive Block-matching Motion Estimation for TV Coding Part II: Inter-frame Prediction,” ***IEEE Trans. Broadcast.***, 3, 102–105, (1991).

[59] Chen, M.C., Willson, A.N., Jr., “A High Accuracy Predictive Logarithmic Motion Estimation Algorithm for Video Coding,” ***in Proc. of IEEE International Symposium on Circuits and Systems (ISCAS’95)***, Seattle, Washington, USA, 1, 223-226, 28 April-3 May, (1995).

[60] Chalidabhongse, J., Kuo, C.C.J., “Fast Motion Vector Estimation using Multiresolution-Spatio-Temporal Correlations,” ***IEEE Trans. Circuits Syst. Video Technol.***, 3, 477–488, (1997).

[61] Tzovaras, D., Strintzis, M.G., Sahinolou, H., “Evaluation of Multiresolution Block Matching Techniques for Motion and Disparity Estimation,” ***Signal Processing: Image Commun.***, 6, 56–67, (1994).

[62] Lee, J.H., Lim, K.W., Song, B.C., Ra, J.B., “A Fast Multiresolution Block Matching Algorithm and its VLSI Architecture for Low Bit-rate Video Coding,” ***IEEE Trans. Circuits Syst. Video Technol.***, 12, 1289– 1301, (2001).

[63] Lee, J.H., Lee, N.S., “Variable Block Size Motion Estimation Algorithm and its Hardware Architecture for H.264,” **in** ***Proc. of IEEE International Symposium on Circuits and Systems (ISCAS’04)***, Vancouver, Canada, 740–743, 23-26 May, (2004).

[64] Li, W., Salari, E., “Successive Elimination Algorithm for Motion Estimation,” ***IEEE Trans. Image Processing***, 1, 105–107, (1995).

[65] Gao, X.Q., Duanmu, C.J., Zou, C.R., “A Multilevel Successive Elimination Algorithm for Block Matching Motion Estimation,” ***IEEE Trans. Image Processing***, 3, 501–504, (2000).

[66] Brunig, M., Niehsen, W., “Fast Full-search Block Matching,” ***IEEE Trans. Circuits Syst. Video Technol.***, 2, 241–247, (2001).

[67] Zhu, C., Qi, W.S., Ser, W., “A New Successive Elimination Algorithm for Fast Block Matching in Motion Estimation,” **in** ***Proc. of IEEE International Symposium on Circuits Systems (ISCAS’04)***, Vancouver, Canada, 733-736, 23-26 May, (2004).

[68] Duanmu, C.J., Ahmad, M.O., Swamy, M.N.S., "8-bit Partial Sum of 16 Luminance Values for Fast Block Motion Estimation," **in *Proceedings of IEEE International Conference on Multimedia Expo (ICME'03)***, Baltimore, USA, 689–692, 6-9 July, (2003).

[69] Digital Video Coding Group, *ITU-T recommendation H.263 software implementation*, ***Telenor R'D***, (1995).

[70] Cheung, C.K., Po, L.M., "Normalized Partial Distortion Search Algorithm for Block Motion Estimation," ***IEEE Trans. Circuits Syst. Video Technol.***, 3, 417–422, (2000).

[71] Kim, J.N., Choi, T.S., "A Fast Full-search Motion-estimation Algorithm using Representative Pixels and Adaptive Matching Scan," ***IEEE Trans. Circuits Syst. Video Technol.***, 7, 1040–1048, (2000).

[72] Lengwehasatit, K., Ortega, A., "Probabilistic Partial distance Fast Matching Algorithms for Motion Estimation," ***IEEE Trans. Circuits Syst. Video Technol.***, 2, 139–152, (2001).

[73] Hatabu, A., Miyazaki, T., Kuroda, I., "Optimization of Decision-timing for Early Termination of SSDA-based Block Matching," ***IEEE International Conference on Acoustic, Speech, and Signal Processing (ICASSP'03)***, Hong Kong, 533–536, 6-10 April, (2003).

[74] Chen, Y.S., Huang, Y.P., Fuh, C.S., "Fast Block Matching Algorithm based on the Winner-update Strategy," ***IEEE Trans. Image Processing***, 8, 1212–1222, (2001).

[75] Srinivasan, S., Hsu, J., Holcomb, T., Mukerjee, K., Regunathan, S. L., Lin, B, Liang, J., Lee, M.-C., Ribas-Corbera, J., "Windows media video 9: overview and application," ***Signal Processing: Image Communication***, 9, 851–875, (2004).

[76] X. Jing, C. Zhu, and L. P. Chau, "Smooth constrained motion estimation for video coding," ***Signal Process.***, 3, 677–680, (2003).

[77] S. Ertürk, "A New Perspective to Block Motion Estimation for Video Compression: High-Frequency Component Matching," ***IEEE Signal Processing Letters***, 2, 113-116, (2007).

[78] Akbulut, O., Urhan, O., Ertürk, S., "Improved Block Motion Estimation using Block Frequency Warping," ***IEEE Signal Processing Letters***, 15, 143-145, (2008).

[79] Kung, S.Y., "*VLSI Array Processors,"* First Edition, ***Prentice Hall***, (1988)

[80] Kung, S.Y., "On supercomputing with systolic/wavefront array processors," ***Proc. IEEE***, 72, 1054-1066, (1984).

[81] Kung, H.T., "Why systolic array," ***IEEE Computers***, 15, 37–46, (1982).

[82] Guan, L., Kung, S.Y., Larsen, J., "Multimedia Image and Video Processing," First Edition, ***CRC Press***, (2001)

[83] Vos, L.D., Stegherr, M., “Parameterizable VLSI Architectures for the Full-search Block-matching Algorithm,” ***IEEE Trans. Circuits Syst.***, 2, 1309–1316, (1989).

[84] Yang, K.M., Sun, M.T., Wu, L., “A Family of VLSI Designs for the Motion Compensation Block-matching Algorithm,” ***IEEE Trans. Circuits Syst.***, 2, 1317–1325, (1989).

[85] Hsieh, C.H., Lin, T.P., “VLSI Architecture for Blockmatching Motion Estimation Algorithm,” ***IEEE Trans. Circuits Syst. Video Technol.***, 2, 169–175, (1992).

[86] Jehng, Y.S., Chen, L.G., Chiueh, T.D., “An Efficient and Simple VLSI Tree Architecture for Motion Estimation Algorithms,” ***IEEE Trans. Signal Processing***, 2, 889–900, (1993).

[87] Chang, S.F., Hwang, J.H., Jen, C.W., “Scalable Array Architecture Design for Full Search Block Matching,” ***IEEE Trans. Circuits Syst. Video Technol.***, 4, 332–343, (1995).

[88] Yeo, H., Hu, Y.H., “A Novel Modular Systolic Array Architecture for Full-search Block Matching Motion Estimation,” ***IEEE Trans. Circuits Syst. Video Technol.***, 5, 407–416, (1995).

[89] Lai, Y.K., Chen, L.G., “A Data-interlacing Architecture with two-Dimensional Data-reuse for Full-search Block-matching Algorithm,” ***IEEE Trans. Circuits Syst. Video Technol.***, 2, 124–127, (1998).

[90] Tuan, J.C., Chang, T.S., Jen, C.W., “On the Data Reuse and Memory Bandwidth Analysis for Full-search Block-matching VLSI Architecture,” ***IEEE Trans. Circuits Syst. Video Technol.***, 1, 61–72, (2002).

[91] Shim, H., Kyung, C.-M., “Data reuse algorithm for multiple reference frame motion estimation,” ***Electronics Letters***, 7, 382 – 383, (2007).

[92] Chen , C.-Y., Huang, C.-T., Chen, Y.-H., Chen L.-G., “Level C+ data reuse scheme for motion estimation with corresponding coding orders,” ***IEEE Transactions on Circuits and Systems for Video Technology***, 4, 553-558, (2006).

[93] Do, V.L., Yun, K.Y., “A Low-power VLSI Architecture for Full-search Block-matching Motion Estimation,” ***IEEE Trans. Circuits Syst. Video Technol.***, 4, 393–398, (1998).

[94] Hanami, A., Scotzniovsky, S., Ishihara, K., Matsumura, T., Takeuchi, S. I., Ohkuma, H., Nishigaki, K., Suzuki, H., Kazayama, M., Yoshida, T., Tsuchihashi, K., “A 165-GOPS Motion Estimation Processor with Adaptive Dual-array Architecture for High Quality Video-encoding Applications,” ***in Proc. of the IEEE Custom Integrated Circuits Conference (CICC’98)***, Santa Clara, California, USA, 169–172, 11-14 May, (1998).

[95] Shen, J.F., Wang, T.C., Chen, L.G., “A Novel Low-power Full Search Block-matching Motion Estimation Design for H.263+,” ***IEEE Trans. Circuits Syst. Video Technol.***, 7, 890–897, (2001).

[96] Roma, N., Sousa, L., “Efficient and Configurable Full-search Block-matching Processors,” ***IEEE Trans. Circuits Syst. Video Technol.***, 12, 1160/1167, (2002).

[97] Huang, Y.W., Wang, T.C., Hsieh, B.Y., Chen, L.G., “Hardware Architecture Design for Variable Block Size Motion estimation in MPEG-4 AVC/JVT/ITU-T H.264,” ***IEEE International Symposium on Circuits Systems (ISCAS’03)***, Bangkok, Thailand, 796–799, 25-28 May, (2003).

[98] Yalcin, S., Ates, H. F., Hamzaoglu, I., "A High Performance Hardware Architecture for an SAD Reuse based Hierarchical Motion Estimation Algorithm for H.264 Video Coding," ***International Conference on Field Programmable Logic and Applications (FPL’2005)***, Tampere, Finland, 509-514, 24-26 August, (2005).

[99] Wei, C., Hui, H., Jiarong, T., Hao, M., “A High-performance Reconfigurable VLSI Architecture for VBSME in H.264,” ***IEEE Trans. on Consumer Electron.***, 3, 1338-1345, (2008).

[100] Jong, H.M., Chen, L.G., Chiueh, T.D., “Parallel Architectures for 3-step Hierarchical Search Block-matching Algorithm,” ***IEEE Trans. Circuits Syst. Video Technol.***, 4, 407–416, (1994).

[101] Dutta, S., Wolf, W., “A Flexible Parallel Architecture Adopted to Block-matching Motion Estimation Algorithms,” ***IEEE Trans. Circuits Syst. Video Technol***., 1, 74–86, (1996).

[102] Lin, H.D., Anesko, A., Petryna, B., “A 14-GOPS Programmable Motion Estimator for H.26x VideoCoding,” ***IEEE J. Solid-State Circuits***, 11, 1742–1750, (1996).

[103] Cheng, S.C., Hang, H.M. “A Comparison of Block matching Algorithms Mapped to Systolic-array Implementation,” ***IEEE Trans. Circuits Syst. Video Technol.***, 5, 741–757, (1997).

[104] Mizuno, M., Ooi, Y., Hayashi, N., Goto, J., Hozumi, M., Furuta, K., Shibayama, A., Nakazawa, Y., Ohnishi, O., Zhu, S.Y., Yokoyama, Y., Katayama, Y., Takano, H., Miki, N., Senda, Y , “A 1.5-W Single-chip MPEG-2 MP@ML Video Encoder with Low Power Motion Estimation and Clocking,” ***IEEE J. Solid-State Circuits***, 11, 1807–1816, (1997).

[105] Takahashi, M., Hamada, M., Nishikawa, T., Arakida, H., Fujita, T., Hatori, F., Mita, S, Suzuki, K., Chiba, A., Terazawa, T., Sano, F., Watanabe, Y., Usami, K., Igarashi, M., Ishikawa, T., Kanazawa, M., Kuroda, T., and Furuyama, T., “A60-mW MPEG-4 Video Codec using Clustered Voltage Scaling with Variable Supply-voltage Scheme,” ***IEEE J. Solid-State Circuits***, 11, 1772–1780, (1998).

[106] Moshnyaga, V.G., “A New Computationally Adaptive Formulation of Block-matching Motion Estimation,” ***IEEE Trans. Circuits Syst. Video Technol***., 1, 118–124, (2001).

[107] Hsia, S.C., “VLSI Implementation for Low-complexity Full search Motion Estimation,” ***IEEE Trans. Circuits Syst. Video Technol.***, 7, 613–619, (2002).

[108] Kawahito, S., Handoko, D., Tadokoro, Y., Matsuzawa, A., “Low Power Motion Vector Estimation using Iterative Search Block-matching Methods and a High-speed Non-destructive CMOS Sensor,” ***IEEE Trans. Circuits Syst. Video Technol.***, 12, 1084–1092, (2002).

[109] Vleeschouwer, C.D., Nilsson, T., Denolf, K. and Bormans, J., “Algorithmic and Architectural Co-design of a Motion estimation Engine for Low-power Video Devices,” ***IEEE Trans. Circuits Syst. Video Technol.***, 12, 1093–1105, (2002).

[110] Chao, W.M., Chen, T.C., Chang, Y.C., Hsu, C.W., Chen, L.G., “Computationally Controllable Integer, Half, and Quarter-pel Motion Estimator for MPEG-4 Advanced Simple Profile,” ***IEEE International Symposium on Circuits Systems (ISCAS’03)***, Bangkok, Thailand, 788–791, 25-28 May, (2003).

[111] Liang, J., Tran, T. D., “Fast multiplierless approximations of the DCT with the lifting scheme,” ***IEEE Trans. Circuit Syst. Video Technol.***, 12, 3032–3044, (2001).

[112] Synopsys, [online], Web adresi: http://www.synopsys.com/, **(Ziyaret Tarihi: 2008)**

[113] Parlak, M., Hamzaoglu, I., “Low Power H.264 Deblocking Filter Hardware Implementations,” ***IEEE Trans. on Consumer Electron.***, 2, (2008).

[114] Baek, Y., Oh, H.S., Lee, H.K., “An efficient block-matching criterion for motion estimation and its VLSI implementation,” ***IEEE Trans. Consumer Electron.***, 4, 885-892, (1996).

[115] Lee, S., Kim, J.M., Chae, S.I., “New motion estimation algorithm using adaptively quantized low bit-resolution image and its VLSI architecture for MPEG2 video encoding,” ***IEEE Trans. Circuits and Syst. Video Technol.***, 6, 734-744, (1998).

[116] Bahari, A., Arslan, T., Erdogan, A.T., “Low power variable block size motion estimation using pixel truncation,” ***IEEE International Symposium on Circuits and Systems (ISCAS’2007)***, New Orleans, USA, 3663-3666, 27-30 May, (2007).

[117] Urhan, O., Ertürk, S., “Blok Bazli Hareket Tahmini Icin Gray-Kodlanmis Imgelerin Bit-Uzaylari Uyulmamasi,” ***10. IEEE Sinyal İşleme ve İletişim Uygulamaları Kurultayı (SIU'2002)***, 518-523, (2002).

[118] Çelebi, A., Urhan, O., Hamzaoğlu, I., Ertürk, S., “Efficient Hardware Implementations of Low Bit Depth Motion Estimation Algorithms,” ***IEEE Signal Process. Letts.***, Submitted for publication, (2008).

[119] Akbulut, O., Urhan, O., Ertürk, S., "Fast Sub-Pixel Motion Estimation by means of One-Bit Transform," Lecture Notes in Computer Science (LNCS), ***Springer Berlin/Heidelberg***, 503-510, (2006).

[120] R. Wang, M. Li, J. Li, Y. Zhang "High Throughput and Low Memory Access Sub-pixel Interpolation Architecture for H.264/AVC HDTV Decoder," ***IEEE Trans. Consumer Electron.***, 3, 1006-1013, (2005).

[121] Yalcin, S., Hamzaoglu, I., "A High Performance Hardware Architecture for Half-Pixel Accurate H.264 Motion Estimation," ***IFIP International Conference on Very Large Scale Integration (VLSI-SoC'2006)***, Nice, France, 63-67, 16-18 October, (2006).

[122] Sühring, K., 2008, H.264/AVC Software Coordination, H.264/AVC JM reference software [online], Web adresi: http://iphome.hhi.de/suehring/tml/, **(Ziyaret Tarihi: 2008)**.

[123] Oktem, S., Hamzaoglu, I., "An Efficient Hardware Architecture for Quarter-Pixel Accurate H.264 Motion Estimation," ***10th Euromicro Conference on Digital System Design Architectures, Methods and Tools (DSD'2007)***, Lübeck, Germany, 444-447, 29-21 August, (2007).

[124] Dias, T., Roma, N., Sousa, L., "Efficient Motion Vector Refinement Architecture for Sub-Pixel Motion Estimation Systems," ***IEEE Workshop on Signal Processing System Design and Implementation (SIPS'2005)***, Athens, Greece, 313-318, 2-4 November, (2005).

KİŞİSEL YAYINLAR ve ESERLER

[1] **Çelebi, A.**, Akbulut, O., Urhan, O., Hamzaoğlu, İ., Ertürk, S., "An All Binary Sub-Pixel Motion Estimation Approach and its Hardware Architecture," ***IEEE Transactions on Consumer Electronics***, 4, (2008). (Yayınlanmak üzere kabul edildi.)

[2] **Çelebi, A.**, Akbulut, O., Urhan, O., Ertürk, S., "Truncated Gray-Coded Bit-Plane Matching Based Motion Estimation and its Hardware Architecture," ***IEEE Trans. Consumer Electron*** (Küçük düzeltmelerle kabul edildi)

[3] **Çelebi, A.**, Urhan, O., Hamzaoğlu, İ., Ertürk, S., "Efficient Hardware Implementations of Low Bit Depth Motion Estimation Algorithms," ***IEEE Signal Process. Lett.*** (Hakem incelemesinde)

[4] **Çelebi, A.**, Urhan, O., Ertürk, S., Hamzaoğlu, İ., Dündar, G., "Kısıtlanmış 1-Bit Dönüşümü Temelli Hareket Kestirimi Algoritmasının HVTDD Yaklaşımıyla Tasarımı," ***IEEE 16. Sinyal İşleme ve İletişim Uygulamaları Kurultayı (SIU'2008)***, Didim, Aydın, 20-22 Nisan, (2008).

[5] **Çelebi, A.**, Urhan, O., Ertürk, S., Dündar, G., "Kısıtlanmış 1-Bit Dönüşümü Temelli Hareket Kestirimi Algoritmasının FPGA Tabanlı Bir Mimari ile Gerçeklenmesi," ***IEEE 15. Sinyal İşleme ve İletişim Uygulamaları Kurultayı (SIU'2007)***, Eskişehir, 1-4, 11-13 Haziran, (2007).

ÖZGEÇMİŞ

1979 yılında Ordu'nun Ünye ilçesinde doğdu. İlköğrenimini, Fatsa Dumlupınar İlkokulu'nda, orta öğrenimini Fatsa Merkez Ortaokulu'nda, lise öğrenimini ise Ordu Fatih Lise'sinde tamamlamıştır. 1998 yılında girdiği Kocaeli Üniversitesi, Mühendislik Fakültesi elektronik ve Haberleşme Mühendisliği Bölümünden 2002 yılında Elektronik ve Haberleşme Mühendisi unvanını ile mezun oldu. 2002 yılında Kocaeli Üniversitesi, Fen Bilimleri Enstitüsünde Yrd.Doç.Dr.Ali TANGEL'in danışmanlığında başladığı yüksek lisans eğitimini 2005 yılında Elektronik ve Haberleşme Yüksek Mühendisi unvanını alarak başarıyla tamamladı. Yüksek lisans eğitiminde hızlı analog sayısal dönüştürücülerin, çok yüksek ölçekte tümleşik devre tasarımı konusunda çalıştı. Bu kapsamda TUBİTAK'ın desteklediği çeşitli araştırma projelerinde araştırıcı olarak yer aldı. 2005 yılında Kocaeli Üniversitesi, Fen Bilimleri Enstitüsünde başladığı doktora eğitimine halen devam etmektedir ve tez aşamasındadır.

2002 yılında Fen Bilimleri Enstitüsü'nde araştırma görevlisi olarak görev yapmaya başlamış olup halen bu görevini yürütmektedir.

Yüksek Lisans Eğitimi Boyunca Gerçekleştirilen Kişisel Yayınlar ve Eserler:

[1] **Çelebi, A.**, Aytar, O., Tangel, A., "A 10-Bit 500Ms/s Two-Step Flash ADC," ***The International Conference on Computer as a Tool (EUROCON'2005)***, Belgrade, Serbia & Montenegro, 898-901, 21-24 November, (2005).

[2] Tekin, M.F., Tangel, A., Aytar, O., **Çelebi A.**, "An 8-Bit CMOS Folding ADC Implementation Using TIQ Based Flash ADC Core," ***12th International Conference Mixed Design of Integrated Circuits and Systems (MIXDES'2005)***, Kraków, Poland, 43-45, 22-25 June, (2005).

[3] Aytar, O., **Çelebi, A.**, Tangel, A., M. F. Tekin, "8-Bit 1GS/s Semi-Flash Adc Based On Threshold Inverter Quantization Technique," ***11th International Conference Mixed Design of Integrated Circuits and Systems (MIXDES'2004)***, Szczecin, Poland, 121-124, 24-26 June, (2004).

[4] **Çelebi**, **A.,** Ertürk, S., Tangel, A., "A VLSI Implementation For Fast All-Boolean Motion Estimation Based On Pre-Coded Image Planes Matching," ***Third International Conference on Electrical and Electronics Engineering, (ELECO'2003)***, Bursa, Turkey, 36-40, 2-7 December, (2003).