KOCAELİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ELEKTRONİK VE HABERLEŞME MÜHENDİSLİĞİ
ANABİLİM DALI

YÜKSEK LİSANS TEZİ

MOSSE NESNE TAKİP ALGORİTMASININ YÜKSEK SEVİYE SENTEZ YAKLAŞIMI İLE FPGA ÜZERİNDE GERÇEKLENMESİ

EMRE TUNÇAY

KOCAELİ 2021

KOCAELİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

ELEKTRONİK VE HABERLEŞME MÜHENDİSLİĞİ

ANABİLİM DALI

YÜKSEK LİSANS TEZİ

MOSSE NESNE TAKİP ALGORİTMASININ YÜKSEK SEVİYE SENTEZ YAKLAŞIMI İLE FPGA ÜZERİNDE GERÇEKLENMESİ

EMRE TUNÇAY

Dr.Öğr.Üyesi Anıl ÇELEBİ		
Danışman,	**Kocaeli Üniversitesi**	**.....**
Prof.Dr. Ali TANGEL		
Jüri Üyesi,	**Kocaeli Üniversitesi**	**.....**
Prof.Dr. Mehmet Kemal GÜLLÜ		
Jüri Üyesi,	**İzmir Bakırçay Üniversitesi**	**.....**

Tezin Savunulduğu Tarih: 22.06.2021

ÖNSÖZ VE TEŞEKKÜR

İnsansız sistemlerin kendisine daha fazla uygulama alanı bulduğu günümüzde, görüntü işleme uygulamaları da paralel olarak önem kazanmaktadır. Bir görüntü işleme uygulamasında, yüksek performans sergileyebilen algoritmalar ve bu algoritmaların gereksinimlerini karşılayabilecek donanımlar gereklidir. Video nesne takip uygulamaları da görüntü işleme uygulamaları arasında önemli bir yere sahiptir.

Video nesne takip uygulamaları, yüksek yoğunluklu verilerin, yüksek hızlı olarak işlendiği, görüntü işleme uygulamalarındandır. Günümüzde yüksek takip performansı gösterebilen çeşitli video nesne takip algoritmaları çalışmaları yapılmaktadır. Bu video nesne takip algoritmalarının, gereksinimlerini karşılayabilecek donanımlar da geliştirilmektedir. FPGA'lar yüksek yoğunluklu veri işleyebilme, paralel çalışabilme ve uygulamaya özel tasarım yapılabilme özellikleriyle birlikte, video nesne takip uygulamalarında tercih edilen donanımlar arasına girmektedir.

Bu tez çalışmasında, MOSSE video nesne takip algoritması, yüksek seviye sentez yaklaşımı ile FPGA donanımı olarak tasarlanmıştır. Çalışmada, yüksek yoğunluklu veri işleyen görüntü işleme algoritmasının FPGA üzerinde gerçeklenmesi ile yüksek çalışma hızlarına ulaşabilmek ve özellikle döner kanatlı hava araçları uygulamalarında önemli gereksinimlerden olan düşük enerji tüketimi ve düşük boyut ihtiyaçlarının karşılanması hedeflenmiştir.

Bu çalışmanın ortaya çıkmasında, fikirleri ve tecrübesi ile bana yol gösteren, bilgi ve düşüncelerini benimle paylaşan, çalışma süresince beni sabırla destekleyen saygıdeğer danışman hocam, Dr.Öğr.Üyesi Anıl ÇELEBİ'ye

Hayatımın her anında bana destek olan, şahsi mutluluğumu ve çıkarımı, kendi mutluluğu ve çıkarları olarak bilen değerli aileme, teşekkür ederim.

Mayıs – 2021 Emre TUNÇAY

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

SİMGELER VE KISALTMALAR DİZİNİ

η	:	Öğrenme katsayısı
p	:	Pearson korelasyon katsayısı

Kısaltmalar

2DFFT	:	Two Dimension Fast Fourier Transform (İki Boyutlu Hızlı Fourier Dönüşümü)
BRAM	:	Block Random Access Memory (Blok Rastgele Erişim Belleği)
DDR	:	Double Data Rate (Çift Veri Hızı)
DMA	:	Direct Memory Access (Doğrudan Bellek Erişimi)
DSST	:	Discriminative Scale Space Tracking (Ayrımcı Ölçek Uzay Takibi)
EOL	:	End of Line (Satır Sonu)
FFT	:	Fast Fourier Transform (Hızlı Fourier Dönüşümü)
FPGA	:	Field Programmable Gate Array (Alan Programlanabilir Kapı Dizileri)
FPS	:	Frame Per Second (Saniye Başına Çerçeve Oranı)
HDMI	:	High Defination Multimedia Interface (Yüksek Çözünürlüklü Multimedya Arayüzü)
HLS	:	High Level Synthesis (Yüksek Seviye Sentez)
HOG	:	Histagram of Oriented Gradients (Yönlendirilmiş Gradyanların Histogramı)
II	:	Initiation Interval (Başlatma Aralığı)
IP	:	Integrated Product (Entegre Ürün)
KCF	:	Kernelized Correlation Filter (Çekirdeklenmiş Korelasyon Filtresi)
MIMO	:	Multi Input Multi Output (Çoklu Giriş Çoklu Çıkış)
MM2S	:	Memory Map to Stream (Bellekten hatta akış)
MOSSE	:	Minimum Output Sum of Square Error (Hataların Karelerinin Toplamının En Küçüğü)
PAR	:	Place and Route (Yerleştir ve Gönder)
PL	:	Program Logic (Programlanabilir mantık devresi)
PS	:	Program System (Programlanabilir Sistem)
RGB	:	Red Green Blue (Kırmızı Yeşil Mavi)
S2MM	:	Stream to Memory Map (Hattan Hafızaya Akış)
SOF	:	Start of Frame (Çerçeve Başlangıcı)
VDMA	:	Video Direct Memory Access (Video Doğrudan Bellek Erişimi)

MOSSE NESNE TAKİP ALGORİTMASININ YÜKSEK SEVİYE SENTEZ YAKLAŞIMI İLE FPGA ÜZERİNDE GERÇEKLENMESİ

ÖZET

Video nesne takip, görüntü işleme uygulamalarının önemli konularındandır. Bir video nesne takip algoritmasından değişken ışık şiddeti, nesne görüntüsünün engellenmesi, nesne boyutu ve yönelimi değişimi gibi durumlarda hareketli nesneleri takip edebilmesi beklenir. Bu gereksinimler dikkate alındığında, yüksek işlem kapasitesine sahip donanımlara ihtiyaç duyulmaktadır. Mobil robotlar, insansız hava araçları gibi uygulamalar dikkate alındığında, yüksek işlem kapasitesine sahip donanımlar güç tüketimi, donanım boyutu gibi etmenlerden dolayı dezavantaj oluşturmaktadır. FPGA (Alanda programlanabilir kapı dizisi) paralel işlem yapabilirlik, büyük verilerin hızlı işlenmesi, tasarıma göre esneklik ve düşük güç tüketimi durumları göze alındığında video nesne takip gibi görüntü işleme uygulamalarında kazanç oluşturmaktadır. Bu çalışmada korelasyon filtresi tabanlı MOSSE (Hata karelerinin toplamının en küçüğü) takip algoritması donanım mimarisi tasarlanmıştır. Tasarlanan donanım mimarisi yüksek seviye sentezleme(YSS) yaklaşımı ile FPGA üzerinde gerçeklenmiştir. Önerilen donanım mimarisinin tek bir FPGA yongası üzerinde gerçeklenmesi, mobil robotlar ve insansız hava araçları gibi faydalı yükün sınırlı olduğu uygulamalarda kazanç sağlayacaktır.

Anahtar Kelimeler: FPGA, Korelasyon Filtresi, MOSSE, Video Nesne Takip, Yüksek Seviye Sentez.

IMPLEMENTATION OF MOSSE OBJECT TRACKING ALGORITHM ON FPGA WITH HIGH LEVEL SYNTHESIS APPROACH

ABSTRACT

Video object tracking is significant topic of image processing aplications. A video tracking algorithm is should to track object in condition, variable light intensity, object occlusion , variable object size and variable object orientation. Considering these requirements, hardware with high processing capacity is needed. Hardware with high processing capacity creates disadvantages in applications such as mobile robot, flying vehicle, due to power consumption and hardware size. FPGA provides high performance in image processing applications by parallelism, processing big data capability and flexibility depending on design. In this paper, hardware architecture of Correlation filter based MOSSE video object tracking algorithm is desinged and designed algorithm is implemented on FPGA with high level synthesis aproach. Implementation of proposed hardware architecture on a single FPGA chip will create advantageous in applications where the payload is limited, such as mobile robots and unmanned aerial vehicles.

Keywords: FPGA, Correlation Filter, MOSSE, Video Object Tracking, High Level Synthesis.

GİRİŞ

Bilim ve teknolojinin gelişmesi ile beraber farklı alanlarda kullanılan teknolojiler otomatik ve akıllı hale getirilmiştir. Özellikle insansız hava araçları ve otonom araçların geliştirilmeye başlanması ile ulaşım, lojistik, savunma ve güvenlik uygulamalarında gelişmiş ve verimli çözümler ortaya çıkmıştır ve beraberinde bu teknolojilerin kullanımı yaygınlaşmıştır.

İnsansız hava araçları, otonom araçlar ve hareketli robotlar insanlı veya insansız olarak kontrol edilebilmekte, kendilerine atanan görevleri icra edebilmektedirler. Bu sistemler kullanım amacına göre görevlerini yerine getirebilmeleri için bulunduğu ortamda yüksek çevresel farkındalığa sahip olabilmesi gerekmektedir. Bu nedenle bu sistemler mekanik, optik, manyetik, elektriksel vb. çeşitli sensörler ile donatılmışlardır.

Günümüzde optik sensör teknolojisi kullanılarak görünür ışık, kızılötesi, termal, hiperspektral kamera teknolojileri geliştirilmiştir. Bu sayede farklı ortam değişkenleri, optik sensörler ve kameralar yardımıyla ölçülebilir hale gelmiştir ve optik sensörler insansız sistemlerde en çok kullanılan sensör teknolojileri arasına girmişlerdir. Kameralar her ne kadar farklı ortam değişkenleri algılayabilseler de bu verilerin işlenebilmesi ve yorumlanabilmesi için bir donanıma ve bu donanım üzerinde çalışan bir sinyal işleme algoritmasına ihtiyaç duymaktadırlar. Bu donanımlar, uygulama gereksinimlerine göre bilgisayar, mikrodenetleyici veya FPGA gibi donanımlar, algoritmalar ise yine uygulama gereksinimlerine göre filtreler, nesne tanıma, nesne takip gibi farklı algoritmalar olabilmektedir. Bu algoritmalar, literatürde görüntü işleme konusu altında incelenmektedir.

Video nesne takip, bir nesnenin konumunu video boyunca takip etme işlemidir. Günümüzde savunma sanayi, trafik uygulamaları, otonom teknolojiler ve robotik gibi alanlarda kullanılmaktadır. Görüntü işleme çalışmalarının artması ile birlikte çeşitli video nesne takip algoritmaları geliştirilmiştir. Bir video nesne takip algoritmasından

ışık şiddeti, nesne oryantasyonu, nesne hızı, kamera titreşimi, nesnenin tamamen veya parçalı engele uğraması gibi çevresel etkilere uyum sağlaması, hızlı çalışabilmesi ve hata oranının düşük olması beklenmektedir. Bu gereksinimlerin karşılanması için yeterli seviyede bir takip algoritması ve bu algoritmayı çalıştırabilecek donanımlara ihtiyaç duyulmaktadır.

Bu tez çalışmasında, korelasyon filtresi tabanlı MOSSE (Hata karelerinin toplamının en küçüğü) video nesne takip algoritması FPGA donanım mimarisi, yüksek seviye sentez yaklaşımı kullanılarak tasarlanmıştır. Önerilen FPGA donanım mimarisinin tek bir FPGA yongası üzerinde gerçeklenmesi, mobil robotlar ve insansız hava araçları gibi faydalı yükün sınırlı olduğu uygulamalarda gerçek zamanlı çalışma, enerji tüketimi ve donanım boyutu özellikleri açısından kazanç sağlayacaktır.

Tez çalışmasında, literatür araştırması bölüm başlığı altında, video nesne takip algoritmaları literatür araştırmalarına yer verilmiştir. Literatür araştırmaları sonucunda video nesne takip algoritmaları sınıflandırılmıştır. MOSSE video nesne takip algoritması, bölüm başlığı altında MOSSE algoritması incelenmiş ve matematiksel modeli açıklanmıştır. Alan programlanabilir kapı dizileri bölüm başlığı altında, FPGA donanımı ve donanım tasarım süreçleri açıklanmıştır. MOSSE video nesne takip algoritması FPGA donanım tasarımı bölüm başlığı altında, tez çalışması kapsamında yapılan FPGA donanım tasarımları, test ve similasyon çalışmaları ve zamanlama analizleri açıklanmıştır. Sonuçlar ve öneriler bölüm başlığı altında, tez çalışması sonucunda elde edilen verilere yer verilmiştir ve ileriye dönük hedeflenen çalışmalar açıklanmıştır.

1. LİTERATÜR ARAŞTIRMASI

Bu bölüm başlığı altında, video nesne takibi, video nesne takip algoritmaları, video nesne takip algoritmaları gereksinimleri ve yapılan çalışmada kullanılan video nesne takip algoritması seçimi anlatılmaktadır.

1.1. Video Nesne Takibi

Video nesne takip, bir kullanıcı veya bir nesne tespit algoritması tarafından seçilen bir nesnenin, ardışık video çerçeveleri boyunca video piksel koordinat düzleminde takip edilmesi işlemidir. Bu işlemi yapan algoritmalara, video nesne takip algoritmaları ismi verilmektedir.

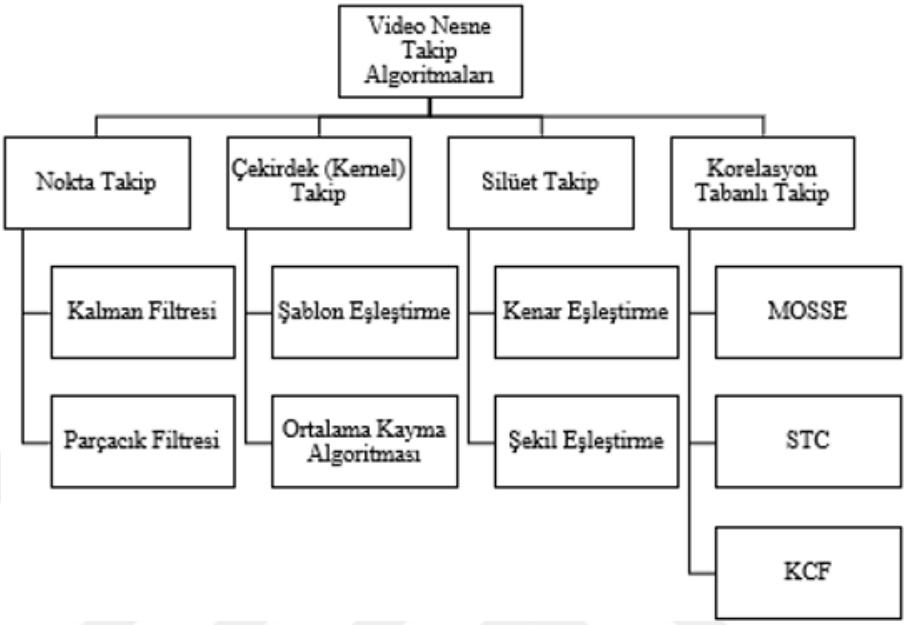
Video nesne takibi, birçok alanda kullanılmaktadır. [1]'de video nesne takip işleminin bazı kullanım alanları verilmiştir.

- Harekete dayalı tanıma: Otomatik nesne tanıma
- Otomatik gözetleme ve güvenlik: Şüpheli hareketlerin izlenmesi
- Video indeksleme: Videoları etiketleme
- İnsan, bilgisayar etkileşimi: Göz hareketlerini tanıma ve takip etme
- Trafik izleme: Trafik akışını izleme ve istatistik tutma
- Araç navigasyonu: Yörünge planlama, engelden kaçınma, yaya ve hedef takibi

1.2. Video Nesne Takip Algoritmaları

Görüntü ve sinyal işleme yöntemlerinin gelişmesi ile birlikte farklı ihtiyaçları karşılayabilen çeşitli video nesne takip algoritması geliştirilmiştir. Bu algoritmaların birbirlerinden üstün ve zayıf yönleri bulunmaktadır. [1]'deki çalışma video nesne takip algoritmaları incelemeleri arasında literatürde saygın bir çalışmadır. [1]'deki çalışmada video nesne takip algoritmaları 3 kategoriye ayrılmıştır. Bunlar nokta takip algoritmaları, kernel takip algoritmaları ve silüet takip algoritmalarıdır.

Bu tez çalışmasında [1]'deki sınıflandırmaya ek olarak, güncel çalışmalarda başarımı yüksek olan korelasyon filtresi tabanlı video nesne takip algoritmaları da eklenmiştir. Video nesne takip algoritmaları sınıflandırmasına Şekil 1.1'de yer verilmiştir.



Şekil 1.1. Video nesne takip algoritmaları sınıflandırması

1.2.1. Nokta tabanlı takip algoritmaları

Nokta tabanlı video nesne takip algoritmaları, video çerçevesi içerisindeki hedef nesne noktalar ile modellemektedir ve hedef nesne konumu kestirimi için nesne ile noktalar arasında ilişkiyi kurmaktadırlar. Nesnelerin noktalar ile ifade edilmesi, nesnenin bir engelle karşılaşması durumunda, nesne takip performansını olumsuz etkilemektedir. Nokta takip algoritmaları deterministtik ve olasılıksal yöntemler olmak üzere ikiye ayrılmaktadır [1].

Deterministtik yöntemler, ardışık imgelerde bulunan nesnelerin birbirleri arasındaki ilişkiyi tanımlamaktadır. Nesneler arasındaki ilişkiyi tanımlamak için yakınlık, maksimum hız, genel hareket gibi sınırlamalar kullanılmaktadır [1].

Olasılıksal yöntemler, nesnenin konum, hız ve ivme değerlerini kullanarak durum uzay modeli oluştururlar [1].

[2]'de W^4 algoritması kullanılmıştır. Çalışmada, nesne arka planları, nesne olmadan birkaç çerçeve boyunca izlenmiş ve model oluşturulmuştur. Oluşturulan model

kullanılarak, video çerçevesine bir nesne girdiğinde nesne takip edilebilmektedir. Çalışmada nesnelerin engele uğraması durumunda nesne takibi yapılabildiği görülmüştür.

İstatistiksel metotlar arasına kalman filtresi tabanlı algoritmalarda girmektedir. [3]'deki çalışmada, hareketli nesnelerin tespiti için değişen ortalama metodu ile arka plandan çıkarma yöntemi kullanılmıştır. Arka planın çıkarılmasından sonra nesne takibi için kalman filtresi kullanılmıştır. Çalışmada, nesne hareketlerinin doğrusal olduğu durumda, kalman filtresinin istenilen sonucu verdiği görülmüştür. Nesneler arasında örtüşme olduğu durumda ise takip başarımı yeterli olmadığı görülmüştür.

1.2.2. Çekirdek tabanlı takip algoritmaları

Çekirdek tabanlı nesne izleyiciler, parametrik olmayan tahmin ediciler gurubu içerisinde yer almaktadır. Parametrik olmayan sistemlerde sabit bir fonksiyon yapısı söz konusu değildir ve bir tahmin gerçekleştirileceği zaman dağılıma ait tüm veri değerleri göz önünde bulundurulmaktadır [4].

Çekirdek tabanlı nesne takibi yönteminin amacı, takip edilecek nesneyi basit bir geometrik şekil içerisine alarak şekil içerisinde kalan görünüm bilgisinin olasılık yoğunluk dağılımını elde etmek ve bu dağılımı ardışık video imgeleri boyunca takip etmektir [4].

[5]'de, çekirdek takip yöntemlerinden biri olan ortalama kayma algoritması kullanılmıştır. Nesnenin takibi için nesnenin histogram değeri hesaplanmıştır ve ortalama kayma algoritması ile nesne takip edilmiştir.

1.2.3. Silüet tabanlı takip algoritmaları

Silüet tabanlı takip algoritmaları, el, parmaklar, insan bedeni gibi belli bir şekil ile ifade edilmesi zor olan ve karakteristik bir geometriye sahip nesnelerin, takip edilmesinde kullanılan algoritmalardır. Siluet tabanlı nesne takip algoritmalarının çalışma mantığı, ardışık görüntü çerçeveleri üzerinde, belirli bir nesne siluetini aramaktır. Silüet tabanlı nesne takip algoritmaları, şekil eşleme ve çevre hatları dönüşümü metotları olarak ikiye ayrılmaktadır [1].

[6]’da şekil eşleme tabanlı takip algoritması kullanılarak, video nesne takibi yapılmıştır . Nesnelerin eşlenmesinde yakın komşuluk yöntemi kullanılmıştır. Bunun yanında hedef nesnenin ağırlık merkezi ve nesne çevresinin renk histogramı nesne takip işleminde kullanılmıştır. Çalışmada, nesnelerin birleşmesi ve birleşen nesnellerin birbirlerinden ayrılması durumuna yönelik bir çözümde önerilmiştir.

1.2.4. Korelasyon filtresi tabanlı takip algoritmaları

Son yıllarda, korelasyon filtresi tabanlı takip algoritmaları üzerine birçok araştırma yapılmıştır. Korelasyon filtresi tabanlı takip algoritmaları, bir kullanıcı veya nesne tespit algoritması ile ilklendirilen ve her görüntü çerçevesinde güncellenen korelasyon filtresi ile nesne takip etme yöntemidir. Korelasyon işlemi zaman uzayından frekans uzayına geçildiğinde nokta çarpım (dot product) çarpım işlemine dönüşmesi, işlem hızına yüksek katkı sağlamaktadır.

[7]’deki çalışmada, hedef nesne ile nesne arka planı arasındaki uzay zaman ilişkisi bayesiyen yapısı ile ifade eden STC algoritması kullanılmıştır. Bu ilişki, hedef nesne ve arka plan arasındaki istatistiksel korelasyonu ifade etmektedir. Kullanılan algoritmanın, nesnenin engellenmesi, ışık şiddeti değişimi ve kamera pozu değişimi gibi çevresel etkilere bağışık olduğu görülmüştür. Ek olarak korelasyon işleminin, frekans uzayında nokta çarpım işlemine dönüşmesi gerçek zamanlı çalışmaya katkı sağlamıştır.

[8]’de dolanır matris, nesne takibinin hızlı bir şekilde yapılabilmesi için Fourier dönüşümü ile diyagonal hale getirilmiştir. Çalışmadaki analitik formülasyon, lineer regresyon ile lineer hale getirildiğinde, korelasyon filtresi yaklaşımına benzediği görülmüştür. Lineer hale getirme işleminde, kernel regresyon kullanılması ile yazar algoritmaya, KCF (Kernelized Correlation Filter) ismini vermiştir. Çalışmada KCF algoritmasının, diğer korelasyon filtresi tabanlı algoritmalara, göre üstünlükleri anlatılmıştır.

[9]’de hedef nesnenin bir kısmının engele uğraması sonucu, takip işleminde oluşan hatalara çözüm olarak, parçalı takip yöntemi KCF filtresi ile birlikte önerilmiştir. Yöntemde, hedef nesne 5 parçaya ayrılmıştır ve her parça ayrı olarak takip edilmiştir.

Bu yöntem ile takip edilen nesnenin kısmı olarak engele uğraması sonucunda, takip performansının olumsuz etkilenmesi durumuna çözüm önerilmiştir.

Korelasyon filtresi tabanlı diğer bir nesne takip algoritması MOSSE algoritmasıdır. [10]'de MOSSE filtresi, korelasyon filtresinin hesaplanmasında kullanılmıştır. Çalışmada, t zamanındaki filtre güncellenirken t-1 zamanındaki filtrede bir öğrenme katsayısı ile hesaba katılmaktadır. Bu sayede nesne oryantasyonu değişse bile filtre bu değişime adapte olabilmektedir. Çalışmada, korelasyon filtresinin frekans uzayında nokta çarpımı haline dönüşümünün algoritma hızına olumsu katkısı ele alınmıştır.

1.3. FPGA Üzerinde Gerçeklenen Çalışmalar

[11]'de korelasyon filtresi tabanlı MOSSE video nesne takip algoritması, Zynq UltraScale+ bir SoC üzerinde gerçeklenmiştir. Çalışmada, korelasyon filtresinin ilklendirilmesi SoC üzerinde bulunan PS tarafında yapılmıştır. Korelasyon filtresinin ilklendirilmesinde, MOSSE algoritmasında bulunan 2DFFT işlemi PS tarafında yazılımsal olarak gerçeklenmiştir. Korelasyon filtresinin ilklendirilmesi dışında kalan işlemler, SoC 'in PL (FPGA) kısmında gerçeklenmiştir. MOSSE algoritmasında bulunan, logaritma ve hanning çerçeveleme işlemleri için LUT kullanılmıştır. Çalışmada PL tarafında kullanılan 2DFFT donanımı, Xilinx FFT IP'si kullanılarak gerçeklenmiştir. 2DFFT donanımının gerçeklenmesi için iki tane tek boyutlu FFT IP'si kullanılmıştır. Çalışmada arama çerçevesi 'nin dinamik değiştirilebilmesinden bahsedilmemiştir. Çalışmada, donanım 300Mhz saat frekansı ile beslendiğinde, 3840×2160 4K çözünürlüğünde bir video'da 64×64 boyutlarındaki bir arama çerçevesinde @60fps ile nesne takip edilmiştir.

[12]'de korelasyon filtresi tabanlı DSST(Discriminative Scale Space Tracking) video nesne takip algoritması, FPGA üzerinde gerçeklenmiştir.

Çalışmada DSST algoritmasının HOG özellik çıkarımı metodu kullanılması ile birlikte MOSSE algoritmasına göre nesne arka plan özelliklerine daha fazla adapte olunduğu belirtilmiştir. Çalışmada DSST algoritmasında kullanılan 2DFFT dönüşümü, 1DFFT dönüşümü donanımı kullanılarak, bir matrisin ilk önce satırları ardından sütunlarının FFT dönüşümü ile yapılmıştır. Çalışmada DSST algoritması Xilinx XC7K325T donanımı kullanılarak @153fps ile hedef takibi yapılmıştır. Çalışmada donanım saat

frekansı ve hangi video çözünürlüğünde belirtilen hedef takip hızına ulaşıldığından bahsedilmemiştir. Çalışmada, hesaplama ihtiyaçlarının görece düşük olması ve nesne takip performansının ise görece yüksek olması nedeniyle DSST algoritması seçilmiştir. Algoritmanın FPGA donanımı üzerinde gerçeklenmesinin sebepleri olarak, yüksek yoğunluklu verinin işlenmese ve yüksek hız isterleri gösterilmiştir.

[13]'de insansız hava araçlarında kullanılmak üzere, tek kart üzerinde çalışabilen bir video nesne takip algoritması hızlandırıcısı tasarlanmıştır. Hesaplama ihtiyaçlarının görece düşük olması ve başarımın görece yüksek olması nedeniyle KCF algoritması seçilmiştir. Hızlandırıcı donanımı olarak, yüksek yoğunluklu verilerin işlenmesi ve hız isterleri göz önüne alındığında, FPGA kullanımına karar verilmiştir. Donanım seçimindeki başka bir kriter olarak, insansız hava araçları gibi enerji tüketiminin darboğaz oluşturduğu sistemlerde, FPGA'in GPU'lara göre düşük güç tüketimine sahip olması etkili olmuştur. Çalışmada FPGA barındıran ZYNQ mimarisi kullanılmıştır. Çalışmada donanım tasarımı için Vivado HLS geliştirme ortamı kullanılmıştır. Vivado HLS ile C++ dili kullanılarak yapılan FPGA donanım tasarımlarının, karmaşık algoritmaların gerçeklenmesinde sağlandığı kazanç üzerinde durulmuştur. Çalışmada KCF algoritmasındaki yüksek yoğunluklu verilerin işlenmesi gereken yerlerde PL tarafı, kontrol, zamanlama ve ilklendirme gereken kısımlarda ise PS tarafı kullanılmıştır. PS ve PL arasındaki haberleşmede AXI arayüzü kullanılmıştır. Çalışma sonucunda tasarlanan KCF algoritması hızlandırıcısı ile ZYNQ Ultrascale+ MPSoC donanımında, 166Mhz frekans saat frekansı ile 960×540 çözünürlüğündeki bir video'da @30fps ile nesne takibi yapılmıştır. Hızlandırıcı, UAV123 veri seti ile test edilmiştir. Çalışmada KCF algoritmasına ek olarak çoklu ölçekleme özelliği de bulunmaktadır. Her video çerçevesinde takip sonucuna göre arama çerçevesi 1.05 oranında büyültme veya küçültülme özelliği eklenmiştir.

[14]'de ikili sınıflandırıcı ve kalman filtresi kullanılarak hesaplama gereksinimleri azaltılan Camshift algoritması, Xilinx Spartan-6 FPGA üzerinde gerçeklenmiştir. Çalışmada, FPGA kaynaklarından tasarruf etmek için YCbCr 4:2:2 video formatı tercih edilmiştir. Geliştirilen algoritmanın doğruluğunu test etmek için MATLAB ortamı kullanılmıştır. Geliştirilen video nesne takip algoritması donanımı ile

148.5Mhz saat frekansı ile çalıştırıldığında HD video'da @309fps hız ile nesne takibi yapılmıştır.

[15]'de video nesne tespit ve video nesne takip algoritmaları, Xilinx Zynq-7000 FPGA üzerinde gerçeklenmiştir. Nesne tespitinde dinamik arka plan çıkarımı, nesne takibinde ise kalman filtresi kullanılmıştır. FPGA donanım tasarımı için Vivado HLS geliştirme ortamı kullanılmıştır. Çalışmada Zynq SoC 'sindeki PS ve PL tarafı beraber kullanılmıştır. Yüksek hesaplama yükünün olduğu kısımlar PL kısmında gerçeklenmiştir. PS ve PL kısımları arasındaki haberleşme için AXI arayüzü kullanılmıştır. Algoritmada bazı işlemlerin donanım olarak gerçeklenmesi PL tarafını karmaşıklaştırması nedeniyle verilerin ön işlenmesi ve donanımlar arasındaki haberleşme için PS tarafının kullanılması tercih edilmiştir. Çalışma PL tarafındaki donanım, 100Mhz saat frekansı ile çalıştırılmıştır ve 1137×686 video çözünürlüğü kullanılmıştır.

[16]'de mobil robot uygulaması için mean-shift video nesne takip algoritması, FPGA kullanılarak gerçeklenmiştir. Çalışmada, video nesne takip algoritması donanım mimarisinde, fazla miktarda çarpma işlemcisi ve bölme işlemcisi bulunmaktadır. Kullanılan Cyclone EP1C6 FPGA'inde algoritma için yeterli olacak çarpma ve bölme işlemcisi olmaması nedeniyle, TDM teknolojisi kullanılarak tasarlanan donanım optimize edilmiştir. Çalışmada geliştirilen video nesne takip algoritması donanımı ile 720×576 video çözünürlüğünde bir video'da @25fps hızında nesne takip edilebilmiştir.

1.4. Video Nesne Takip Algoritmaları Gereksinimleri

Tez çalışması kapmasında, video nesne takip algoritmaları için literatür araştırmaları yapılmıştır. Literatür araştırması sonucunda, video nesne takip algoritmaları, dört sınıf altında incelenmiştir. Literatürde birçok video nesne takip algoritması çalışması bulunmaktadır. Geliştirilen her algoritma, bir gereksinime çözüm olmayı hedeflemiştir. Bu doğrultuda, bir video nesne takip algoritmasından beklenen gereksinimlere Tablo 1.1'de yer verilmiştir.

Tablo 1.1. Video nesne takip algoritması gereksinimleri

Gereksinim	Gereksinimin Tanımı
1	Gürültü, ışık miktarı değişimi, karmaşık nesne şekli, hareketli arka plan çevresel şartlara bağışık olmalı.
2	Nesne görüntüsü kısmı olarak engele uğradığında nesne takibine devam edebiliyor olmalı.
3	Nesne hareketine ve oryantasyonuna göre algoritma adapte olabiliyor olmalı.
4	Hedef nesne'nin merkez koordinatını yüksek doğruluk payı ile takip edebiliyor olmalı.
5	Gerçek zamanlı çalışma gereksinimlerini karşılıyor olmalı.

2. MOSSE VİDEO NESNE TAKİP ALGORİTMASI

Bu konu başlığı altında, korelasyon filtresi tabanlı MOSSE video nesne takip algoritmasının matematiksel modellemesi yapılmaktadır ve algoritmanın çalışması açıklanmaktadır.

2.1. Korelasyon Filtresi Tabanlı Video Nesne Takip

Filtre tabanlı takip algoritmaları, hedef nesne görüntüsü ile eğitilmiş filtreleri kullanarak nesne takip etmektedir. Hedef nesnenin merkezde bulunduğu bir arama penceresi (ROI), filtre ilklendirilirken seçilmektedir. Nesne takibi sırasında ise filtre güncellenmektedir. Elde edilen filtre, hedef nesnenin bir sonraki görüntü çerçevesinde aranması için kullanılmaktadır.

$$f = g \times h \tag{2.1}$$

Denklem (2.1)'de korelasyon filtresi ifade edilmektedir. Korelasyon matrisi g, arama penceresi f, korelasyon filtresi h olarak ifade edilmektedir.

Wiener–Khinchin teoremine göre, korelasyon işlemi zaman uzayından frekans uzayına geçildiğinde, eleman bazlı çarpım işlemine dönüşmektedir. Korelasyon işleminin, eleman bazlı çarpım işlemine dönüşmesi, daha az matematiksel işlem ile korelasyonun hesaplanmasına olanak sağlamaktadır. Bu dönüşüm korelasyon filtresi tabanlı takip algoritmalarında gerçek zamanlı çalışma gereksinimin sağlanması için fayda sağlamaktadır. Denklem (2.1)'de zaman uzayından frekans uzayına geçildiğinde, Denklem (2.2)'deki gibi ifade edilmektedir.

$$G = F \odot H^{*} \tag{2.2}$$

Denklem (2.2)'de korelasyon matrisinin frekans uzayında gösterimi G, arama penceresinin frekans uzayında gösterimi F ve korelasyon filtresinin frekans uzayında gösterimi H'dir.

Elde edilen korelasyon matrisinin zaman uzayına dönüşümü sağlandığında korelasyonun en yüksek olduğu nokta hedef nesnenin merkez noktası olarak kabul edilmektedir.

2.2. MOSSE Filtresi

MOSSE filtresi, nesnenin merkezini işlemin çıkışına taşıyacak en uygun filtreyi bulmayı amaçlayan bir yöntemdir. Denklem (2.2)'den faydalanılarak korelasyon filtresi, Denklem (2.3)'deki gibi ifade edilmektedir.

$$H_i^* = \frac{G_i}{F_i} \tag{2.3}$$

Hedef takibinde G_i 'nin hedef nesneyi merkez alan 2 boyutlu bir gausyen dağılım olması beklenmektedir. MOSSE filtresi, en iyi korelasyon filtresinin bulunması için denklem (2.4) tanımlamaktadır.

$$\min_{H^*} \sum_i \left| F_i \odot H_i \right| \tag{2.4}$$

g_i matrisindeki tepe noktası, f_i arama penceresinde hedef nesneyi merkez almaktadır. Hedef takibinde, hedef nesne her zaman matrisin merkezinde olmayacaktır bu nedenle g_i matrisi bir çok şekilde olabilmektedir. Bu durumu optimize edebilmek için MOSSE Denklem (2.5) tanımlanmaktadır.

$$0 = \frac{\partial}{\partial H_{wv}^*} \sum \left| F_{iwv} H_{wv}^* - G_{iwv} \right|^2 \tag{2.5}$$

Denklem (2.5)'de bulunan diferansiyel denklem çözüldüğünde, Denklem (2.6)'daki MOSSE filtresi elde edilmektedir.

$$H^* = \frac{\sum_i G_i \odot F_i^*}{\sum_i F_i \odot F_i^*} \tag{2.6}$$

2.3. MOSSE Filtresinin İlklendirilmesi ve Güncellenmesi

Takip edilen hedef nesne, video içerisinde zamanla dönebilir, kameraya yaklaşıp uzaklaşabilir. Bu gibi değişken etkiler oluşturulan korelasyon filtresinin bozulmasına sebep olmaktadır. Bu duruma çözüm olabilmesi için korelasyon filtresinin değişken çevresel şartlara adapte olacak şekilde güncellenmesi gerekmektedir. MOSSE algoritmasının tanımladığı filtre denklem (2.7)'de verilmiştir.

$$H_i^* = \eta \frac{G_i \odot F_i^*}{F_i \odot F_i^*} + (1 - \eta) H_{i-1}^* \tag{2.7}$$

MOSSE algoritması, t zamanındaki korelasyon filtresini güncellerken, t-1 zamanındaki korelasyon filtresini, bir öğrenme katsayısı(η) ile ağırlıklandırarak kullanmaktadır.

$$H_i^* = \frac{A_i}{B_i} \tag{2.8}$$

Korelasyon filtresi, Denklem (2.8)'deki gibi A_i ve B_i bileşenleri şeklinde ifade edilebilir. Denklem (2.7) denklem (2.8) gibi yazıldığında, A_i ve B_i bileşenleri sırasıyla denklem (2.9) ve denklem (2.10) tanımlanmaktadır.

$$A_i = \eta G_i \odot H_i + (1 - \eta) A_{i-1} \tag{2.9}$$

$$B_i = \eta F_i \odot F_i^* + (1 - \eta) B_{i-1} \tag{2.10}$$

Korelasyon filtresi A_i ve B_i bileşenleri, Denklem (2.9) ve Denklem (2.10) kullanılarak, her görüntü çerçevesinde güncellenmektedir.

Takip işleminin ilk aşamasında, takip algoritmasına, hedef nesnenin ilk konumu, bir kullanıcı veya bir nesne tanıma algoritması tarafından verilmektedir. Verilen ilk nesne konumu ile korelasyon filtresi ilklendirilmektedir. Korelasyon filtresi ilklendirilmesi, filtre güncelleme adımlarına benzerlik göstermektedir. Filtre ilklendirme aşamasında henüz G_i korelasyon matrisi oluşturulmadığından, hedef nesne konumunu tepe noktası olarak alan, bir 2 boyutlu gausyen matrisi G_i matrisi kullanılmaktadır.

2.4. Ön İşlem

Görüntü çerçevesinin, korelasyon işleminde kullanılmak üzere zaman uzayından frekans uzayına geçişinde, sinyale spektral gürültü eklenmektedir. Bu spektral gürültü korelasyon çıkışında yapay bir etkiye sebebiyet vermekte ve olumsuz etki yaratmaktadır. Fourier dönüşümünde olan spektral sızıntıya çözüm olmak için görüntü çerçevesi f_i 'ye bir ön işlem yapılmaktadır. Denklem (2.11)' ön işlem için kullanılan denklem verilmektedir.

$$N(f_i) = \frac{\log(f_i) - \bar{x}_{f_i}}{\sigma_{f_i}} \tag{2.11}$$

Denklem (2.11) kullanılarak normalize edilen görüntü matrisi, 2 boyutlu çerçeve fonksiyonu ile çarpılmaktadır.

2.4.1. Çerçeve fonksiyonu

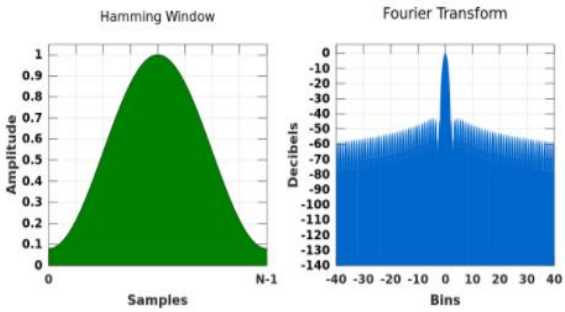
Çerçeve fonksiyonları, FFT dönüşümü sırasında oluşan spektral gürültüleri engellemek için kullanılan fonksiyonlardır. Sinyal işleme uygulamalarında, FFT dönüşümünden önce spektral gürültüyü önlemek için sinyal bir çerçeve fonksiyonu ile çarpılmaktadır. Matematikçiler çeşitli çerçeve fonksiyonları türetmişlerdir. Çerçeve fonksiyonları, uygulamaya göre avantaj ve dezavantaj göstermektedirler. Çerçeve fonksiyonu seçiminde MLW, SLH ve SLRR parametreleri dikkate alınmaktadır.

İdeal bir çerçeve fonksiyonunda

- MLW parametresinin düşük olması.
- SLH parametresinin düşük olması.
- SLRR parametresinin büyük olması beklenmektedir.

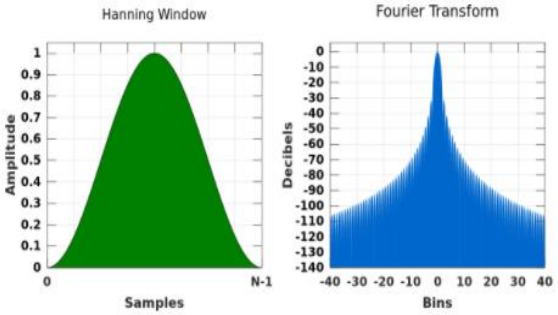
Sinyal işleme uygulamalarında hamming ve hanning çerçeve fonksiyonları tercih edilen çerçeve fonksiyonlarıdır. Barlett çerçeve fonksiyonu daha az tercih edilen bir çerçeve fonksiyonudur. Hamming, hanning ve barlett çerçeve fonksiyonu dışında çeşitli çerçeve fonksiyonlarıda bulunmaktadır.

MLW: 8π/N
SLH: -43.7547 dB
SLRR: -6 dB/octave
ENBW: 1.3631



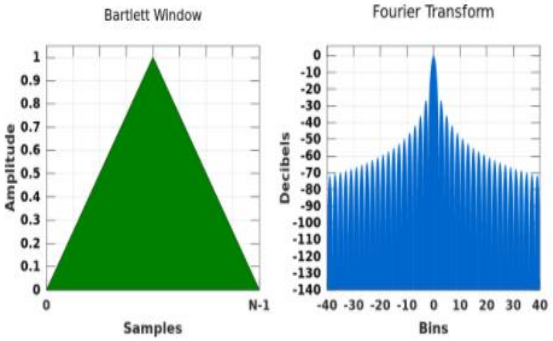
Şekil 2.1. Hamming çerçeve fonksiyonu [17]

MLW: 8π/N
SLH: -31.5565 dB
SLRR: -18 dB/octave
ENBW: 1.5



Şekil 2.2. Hanning çerçeve fonksiyonu [17]

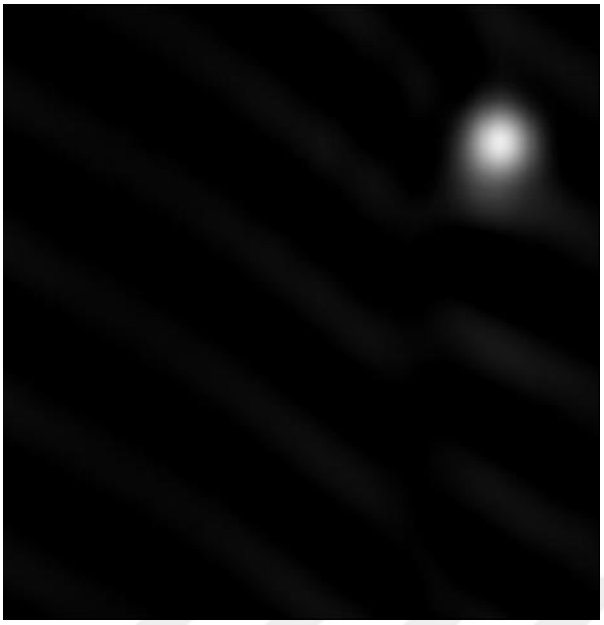
MLW: 8π/N
SLH: -26.5269 dB
SLRR: -12 dB/octave
ENBW: 1.333



Şekil 2.3. Barlett çerçeve fonksiyonu [17]

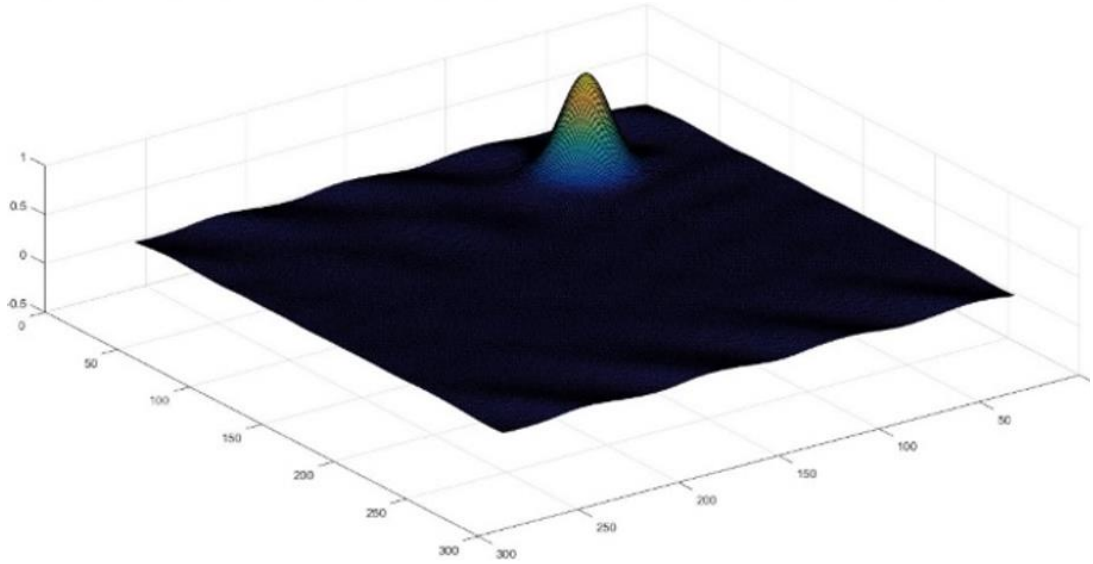
2.5. Hedef Nesne Konumunun Kestirilmesi

Güncellenmiş korelasyon filtresi ile arama penceresinin, Denklem (2.2)'deki gibi eleman bazlı çarpılması sonucunda, korelasyon matrisi elde edilmektedir. Hedef nesne, görüntü çerçevesinde bulunuyorsa korelasyon matrisi iki boyutlu bir gaussyen dağılıma benzemektedir.



Şekil 2.4. 2 boyutlu gausyen dağılım gösteren korelasyon matrisi

Elde edilen gausyen dağılımın 3 boyutlu gösterimi Şekil 2.5'de gösterilmektedir. Gausyen dağılımın tepe noktası, hedef nesnenin yeni merkez noktasının, piksel koordinat düzlemindeki koordinatlarını ifade etmektedir.



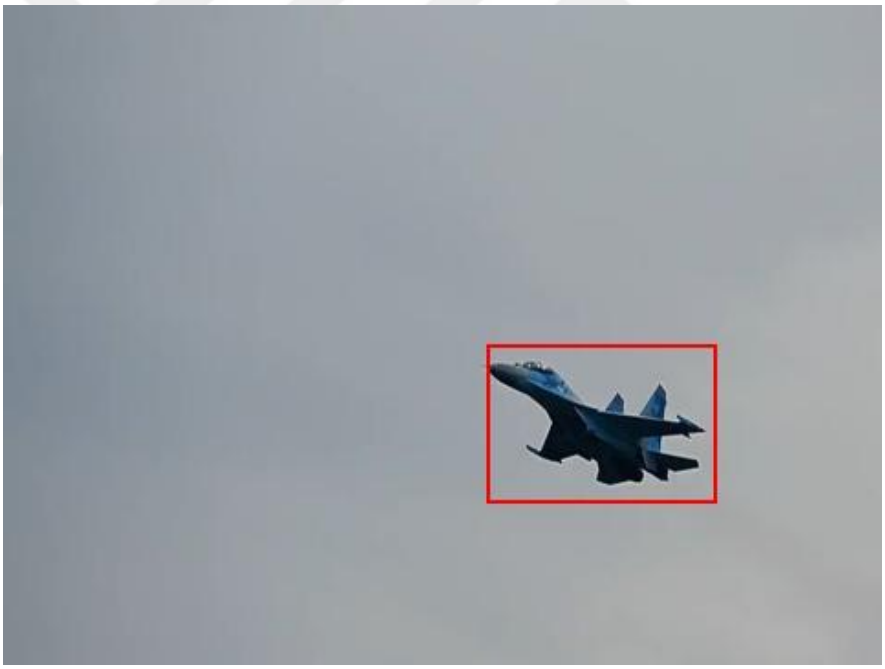
Şekil 2.5. Elde edilen korelasyon matrisinin 3 boyutlu gösterimi

2.6. Arama Penceresi

Gerçek zamanlı bir videodaki bir nesnenin, ardışık iki video çerçevesindeki konumları istatistiksel olarak birbirine çok yakın kabul edilmektedir. Bu kabul 'den yola

çıkılarak, bir nesnenin t+1 zamanındaki video çerçevesindeki konumu, orta noktası t zamanındaki video çerçevesindeki konumu kabul edilen bir arama penceresi içerisinde aranabilir. Arama penceresi genellikle video çözünürlüğünden düşük seçilmekte ve nesnenin boyutuna ve kameraya olan uzaklığına göre seçilmektedir.

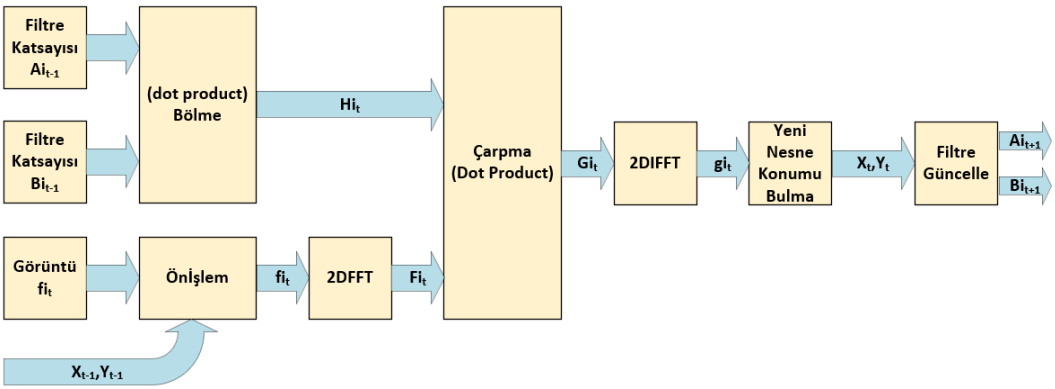
Denklem (2.3)'de bulunan H_i, F_i ve G_i bileşenleri, boyutları arama penceresi boyutlarında olan birer matristir. Arama penceresinin gereksiz yere büyük seçilmesi, MOSSE algoritmasındaki matris boyutlarını arttıracak, dolayısıyla hesaplama gecikmelerini arttıracaktır. Arama penceresinin küçük seçilmesi, nesnenin kameraya yakınlaşması ve kapladığı alanın artması ile birlikte nesnenin boyutu arama penceresi boyutundan büyük olmasına sebep olacaktır. Bu durum nesne takip performansını olumsuz yönde etkilemektedir. Her iki durum dikkate alındığında arama penceresi boyutu uygun bir boyutta seçilmelidir.



Şekil 2.6. Arama penceresi çerçevelenmiş nesne görüntüsü

2.7. MOSSE Video Nesne Takip Algoritması Blok Diyagramı

Matematiksel modeli açıklanan MOSSE video nesne takip algoritmasının blok diyagramı, Şekil 2.7'de verilmiştir. MOSSE video nesne takip algoritması blok diyagramı, FPGA donanım mimarisi tasarımı aşamasında referans alınmıştır. MOSSE video nesne takip algoritması Algoritma:1 tablosunda tanımlanmıştır.



Şekil 2.7. Mosse video nesne takip algoritması blok diyagramı

Algoritma 1: MOSSE Video Nesne Takip Algoritması

```
1   Başla
2   Nesne ilk konumunu ve arama penceresi boyutunu al
3   Ai ve Bi filtre katsayı matrislerini ilklendir
4   WHILE
5       Görüntü çerçevesi matrisinden "fi" arama penceresini çıkar
6       fi arama penceresine önişlem yap
7       fi arama penceresini 2DFFT dönüşümü ile frekans boyutuna taşı Fi
        matrisini elde et
8       Hi korelasyon filtresi matrisini hesapla
9       Gi korelasyon matrisini hesapla
10      Gi korelasyon matrisinin tepe noktasını bul, nesne konumu kestirimi yap
11      Ai ve Bi korelasyon filtresi katsayı matrislerini güncelle
12  END
```

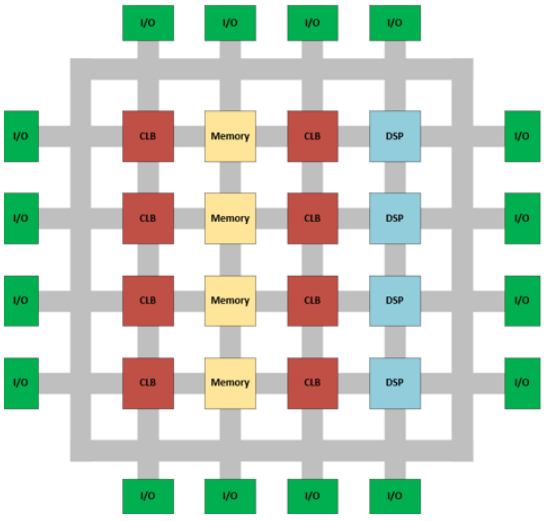
3. ALAN PROGRAMLANABİLİR KAPI DİZİLERİ (FPGA)

Alan programlanabilir kapı dizileri, imalattan sonra farklı algoritmalar için tekrardan programlanabilen, entegre devrelerdir. FPGA'ler, çeşitli yazılım algoritmalarını uygulamak için yapılandırılabilen, iki milyona kadar mantık hücresinden oluşur. Geleneksel FPGA donanım tasarım akışı, bir işlemciden çok normal bir entegre devreye benzese de, bir FPGA bir entegre devre geliştirme çabasına kıyasla önemli maliyet avantajları sağlamaktadır ve çoğu durumda aynı seviyede performans sunabilmektedir. FPGA'in entegre devre ile karşılaştırıldığında bir başka avantajı ise dinamik olarak yeniden yapılandırılabilmesidir.

3.1. FPGA İç Yapısı

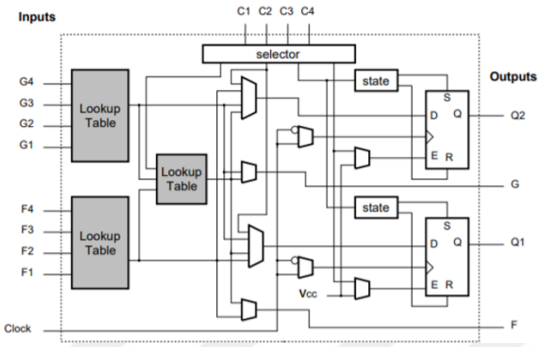
FPGA'ler Blok ram'leri, DSP işlem birimlerini, PCI Express desteği ve programlanabilir mantık devrelerini içeren, heterojen hesaplama platformlarıdır. Bu donanımların hepsi aynı anda kullanılabildiği için paralel işlem yapabilme kabiliyetine sahiptirler [17].

FPGA'ler temelde üç bloktan oluşmaktadır. Bunlar lojik bloklar(CLB), giriş-çıkış blokları(I/O) ve bağlantı bloklarıdır.



Şekil 3.1. FPGA İç Yapısı

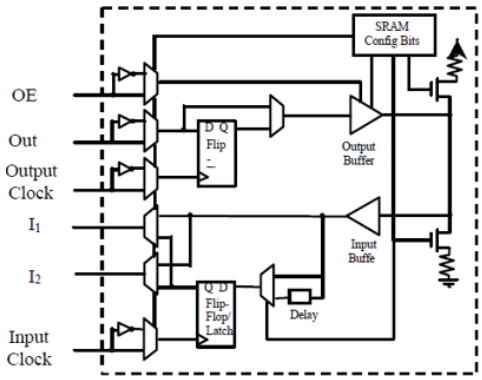
Lojik Bloklar(CLB): Şekil 3.2'de gösterilen lojik bloklar, mantıksal işlemlerin gerçeklendiği yapılardır. Küçük taneli ve kaba taneli olmak üzere iki sınıfa ayrılırlar. Bu sınıflandırmada, lojik bloğun üretiminde kullanılan transistör sayısı, lojik bloğun gerçekleyebileceği mantıksal fonksiyon sayısı ve lojik bloğun giriş-çıkış sayısı büyüklük ölçütü olarak kullanılmaktadır [19].



Şekil 3.2. FPGA mantık hücresi[20]

Bağlantı Blokları: Lojik bloklar ile giriş-çıkış blokları arasındaki bağlantıyı sağlayan yapılardır. Bu yapılar, yollandırma kanalları ve programlanabilir anahtarlardan oluşmaktadır. FPGA mimarileri, bağlantı kanallarının yapısına göre simetrik dizi mimarisi, sıra tabanlı mimari ve kapı denizi mimarisi olmak üzere üç gruba ayrılmaktadır [19].

Giriş Çıkış Blokları: FPGA'lerin programlanabilir giriş-çıkış terminalleridir. Bu giriş-çıkış bloklarında yer alan pinler isteğe bağlı giriş, çıkış veya hem giriş hem çıkış olarak yapılandırılabilmektedir. Bu bloklar, dış dünya ile çip içindeki donanım arasında köprü vazifesi görmektedir.

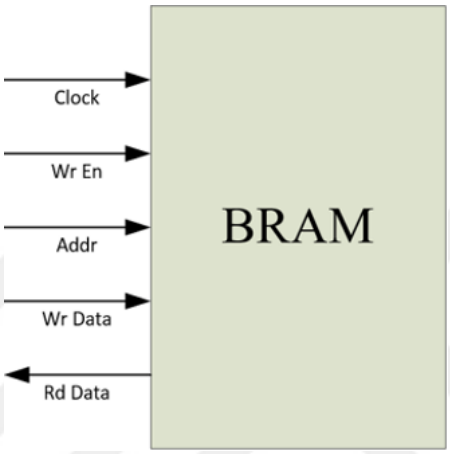


Şekil 3.3. FPGA giriş-çıkış bloğu[21]

3.2. FPGA Hafıza Yapıları

FPGA'ler, üzerinde gerçeklenen lojik devrelerde kullanılmak üzere farklı hafıza elemanları barındırmaktadırlar. Rastgele erişim belleği (RAM-BRAM), salt okunur bellek (ROM), kaydedici (register), kaydırmalı yazmaçlar (shift register) ve FIFO bu hafıza elemanlarından bazılarıdır.

BRAM, nispeten büyük veri kümelerinin depolanması kullanılan hafıza elemanıdır.



Şekil 3.4. BRAM Blok Diyagramı

Kayan yazmaç, arka arkaya birbirine bağlanmış kaydedici (register)'lardan oluşan hafıza elemanıdır. Nispeten küçük verileri depolamak için kullanılmaktadırlar.

FIFO (ilk giren ilk çıkar), tek giriş ve tek çıkış portu bulunan, verilerin içerisine yazılma sırasına göre kayıt edildiği, okunacağı zaman yine kayıt edilme sırasına göre okunduğu bir hafıza elemanıdır. Belli bir hattan gelen verileri, kayıt etmek ve işleme sırasında eş zamanlılığı sağlamak amaçlı kullanılmaktadırlar.

Şekil 3.5. FIFO Hafıza Elemanı

Bu hafıza elemanları dışında FPGA uygulamalarında DDR, Flash hafıza gibi dış kaynaklı hafıza elemanlarını da kullanırlar. Tasarlanan algoritmanın, nispeten büyük hafıza elemanlarına ihtiyaç duyması FPGA maliyetini arttırmaktadır. Bu nedenle FPGA uygulamalarında, dış kaynaklı hafıza elemanlarının kullanımı tercih edilebilmektedir.

3.3. FPGA Uygulamaları

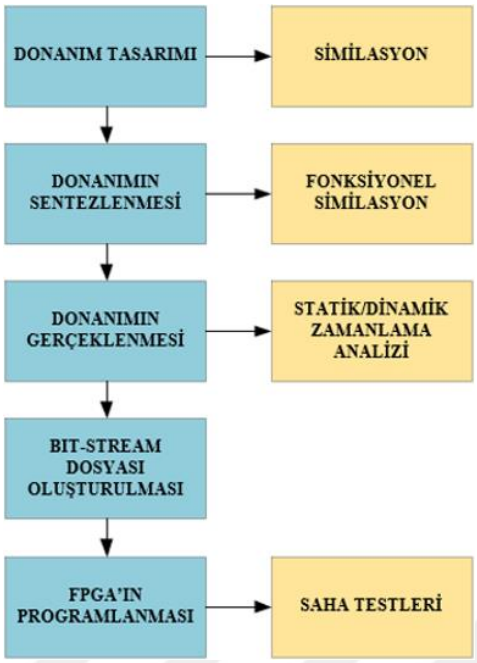
FPGA'ler, tasarım aşamasında tasarımcıya, esneklik sağlaması ve paralel işlem yapabilmeleri sebebiyle bir çok alanda kullanılmaktadırlar. Aşağıda FPGA'lerin bazı kullanım alanları verilmiştir.

- Havacılık ve Savunma sanayi
- Otomotiv
- Haberleşme
- Tüketici Elektroniği
- Veri Merkezleri
- Yüksek hızlı hesaplama uygulamaları
- Biyomedikal
- Test ve ölçüm uygulamaları
- Sinyal işleme uygulamaları
- Video ve görüntü işleme uygulamaları

FPGA'ler, uygulamaya özel tasarım esnekliği, yüksek hız gereksinimlerini karşılama, güvenlik, düşük güç tüketimi gibi avantajlarının yanında, tasarım süreçlerin donanım tasarımı tecrübesi ve mühendislik süreçlerinin uzun olması nedeniyle dezavantajda oluşturabilmektedir. Bazı FPGA üreticileri, bu dezavantajlara çözüm sunabilmek için yüksek seviye sentezleme araçları geliştirmektedir.

3.4. FPGA Donanım Tasarım Adımları

FPGA donanım tasarımında, izlenen farklı tasarım adımları vardır. Bu tasarım adımları tasarıma ve tasarımcıya göre farklılık göstermekle beraber genel olarak Şekil 3.6'daki gibi ifade edilmektedir.



Şekil 3.6. FPGA donanım tasarım adımları

3.4.1. Donanım tasarımı

Donanım tasarımı adımı, FPGA donanım tasarımının ilk adımıdır. Bu adımda, uygulamanın gereksinimlerine göre sayısal devre tasarımı yapılır. Tasarlanan sayısal devre, Verilog, VHDL gibi donanım tanımlama dilleri ile tanımlanabileceği gibi şematik, blok dizayn veya yüksek seviye sentez araçları ile tanımlanabilir. Sayısal devre tasarımı, uygulanılacak algoritmaya hakim olmanın yanında, yüksek sayısal devre tasarımı tecrübesi gerektirmektedir.

3.4.2. Donanımın sentezlenmesi

Bu adımda, donanım tasarım adımında tasarlanan ve donanım tanımlama dilleri ile tanımlanan donanım, kapı seviyesinde devrelere dönüştürülür.

Sentezleme, donanım tanımlama dili ile yazılan kodun kurallarının kontrolü ile başlamaktadır. Ardından lojik devrenin küçültülmesi ve optimize edilmesi ile devam eder. Son olarak tasarım bir sayısal devreye dönüştürülmektedir. Sentezleme işlemi sonunda, bir netlist dosyası üretilmektedir. Sentezleme, özel sentezleyici yazılım araçları ile yapılmaktadır. Bu sentezleyici araçlar, FPGA üreticileri tarafından üretileceği gibi farklı firmalar tarafından da üretilmektedir. Xilinx firmasının XST ve

Vivado, Intel Altera firmasının Quartus 2 ve Lattice firmasının IspLever araçları bazı sentezleyici araçlarındandır.

3.4.3. Donanımın gerçeklenmesi (Implementation)

Bu adım, dönüşüm, eşleştirme, ve PAR (translate, map ve place and route) aşamalarından oluşmaktadır. Dönüşüm aşamasında, tasarımcı tarafından tanımlanan kısıtlar netlist dosyası ile birleştirilmektedir. Bu kısıtlar, I/O(giriş/çıkış) pinlerinin konumları, saat frekansı ve gecikmeler gibi değişkenler olmaktadır. Eşleştirme aşamasında, tasarlanan lojik devre alt bloklara bölünür ve hedef FPGA donanımı üzerinde haritalanır. PAR aşamasında, eşleştirme aşamasında oluşturulan tüm alt blokları, I/O pinlerini ve tüm sinyalleri birbirine bağlanmaktadır [22].

3.4.4. Similasyon ve zamanlama analizi

FPGA donanım tasarım adımlarının, tasarımın doğrulanması amacıyla simülasyonlar ve zamanlama analizleri yapılmaktadır. Simülasyonlar davranışsal ve fonksiyonel similasyon olarak ikiye ayrılmaktadır. Davranışsal similasyon, sentezleme adımından önce yapılmaktadır. Fonksiyonel similasyon ise sentezleme adımından sonra yapılmaktadır ve sentezlenen sayısal devrenin, amaçlanan fonksiyonelliğin sağlandığı doğrulanmaktadır.

Zamanlama analizi, tasarlanan sayısal devrenin zaman gecikmelerinin hesaplandığı ve sayısal devrenin hangi saat frekansları ile çalışabileceğinin belirlendiği analizdir. Zamanlama analizi sonucunda, zamanlama gereksinimlerini karşılayaman tasarım için donanım tasarım adımına geri dönülmektedir. Zamanlama analizi statik ve dinamik zamanlama analizi olarak ikiye ayrılmaktadır.

3.5. Yüksek Seviye Sentezleme

Günümüzde görüntü işleme, yapay zeka, kontrol ve sinyal işleme algoritmalarının, teknolojinin gelişmesi ile beraber karmaşıklığı ve işlediği veri boyutu artmaktadır ve yüksek hızlarda çalışabilme gereksinimi her geçen gün önem kazanmaktadır. Bu gereksinimler ile birlikte, FPGA donanım tasarım süreçlerindeki mühendislik çalışmaları karmaşıklaşmakta ve adam/saat oranı artmaktadır. FPGA üreticileri bu duruma çözüm olarak, yüksek seviye sentezleme araçları geliştirmişlerdir. Yüksek

seviye sentez araçları ile birlikte geliştirici C/C++ ve benzeri diller ile tasarım yapabilmektedir. Yüksek seviye sentezleme araçları ile FPGA donanım tasarımında geliştirici göreceli olarak VHDL ve Verilog gibi donanım tanımlama dillerine göre donanımdan soyutlanmaktadır.

4. MOSSE VİDEO NESNE TAKİP ALGORİTMASI FPGA DONANIM TASARIMI

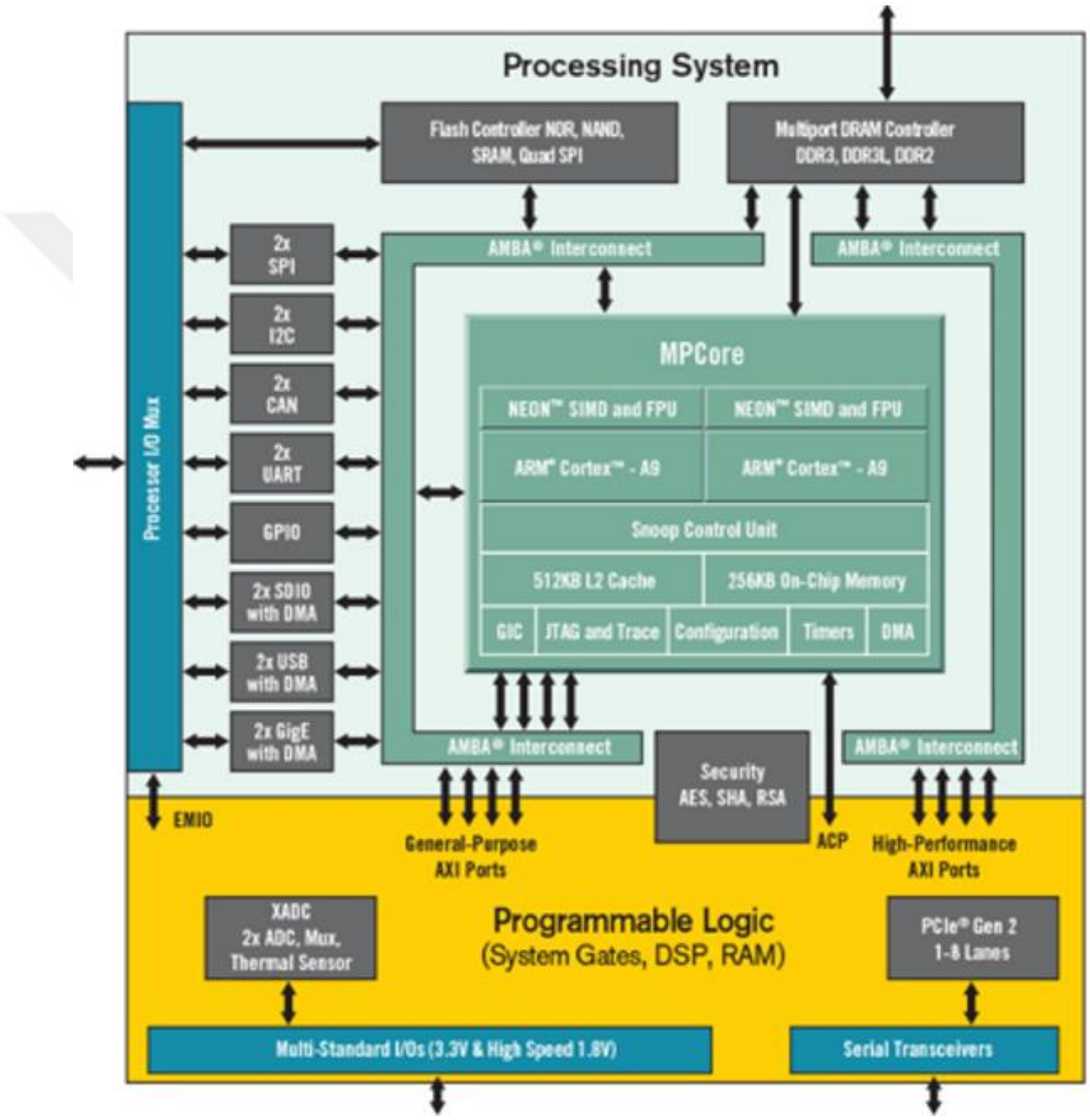
Bölüm 2'de açıklanan MOSSE algoritması, MATLAB yazılımı kullanılarak yazılımsal olarak gerçeklenmiştir. MATLAB ortamında yazılımsal olarak gerçeklenen ve doğrulanan MOSSE video nesne takip algoritması, Şekil 2.7'deki blok şemada olduğu gibi alt işlem adımlarına ayrılmıştır. Ayrılan her bir alt işlem adımı için FPGA donanım mimarisi tasarlanmıştır ve Vivado HLS ortamında, yüksek seviye sentez yaklaşımı ile FPGA donanımı olarak tasarlanmıştır.

Bu bölüm başlığı altında, bölüm 2'de matematiksel altyapısı açıklanan ve Şekil 2.7'deki gibi algoritma blok şeması oluşturulan, MOSSE video nesne takip algoritmasının, FPGA donanım mimarisi, FPGA donanım tasarımı çalışmaları, donanım arayüzleri, similasyon ve zamanlama analizleri açıklanmaktadır.

4.1. Nesne Takip Edici Donanım Mimarisi

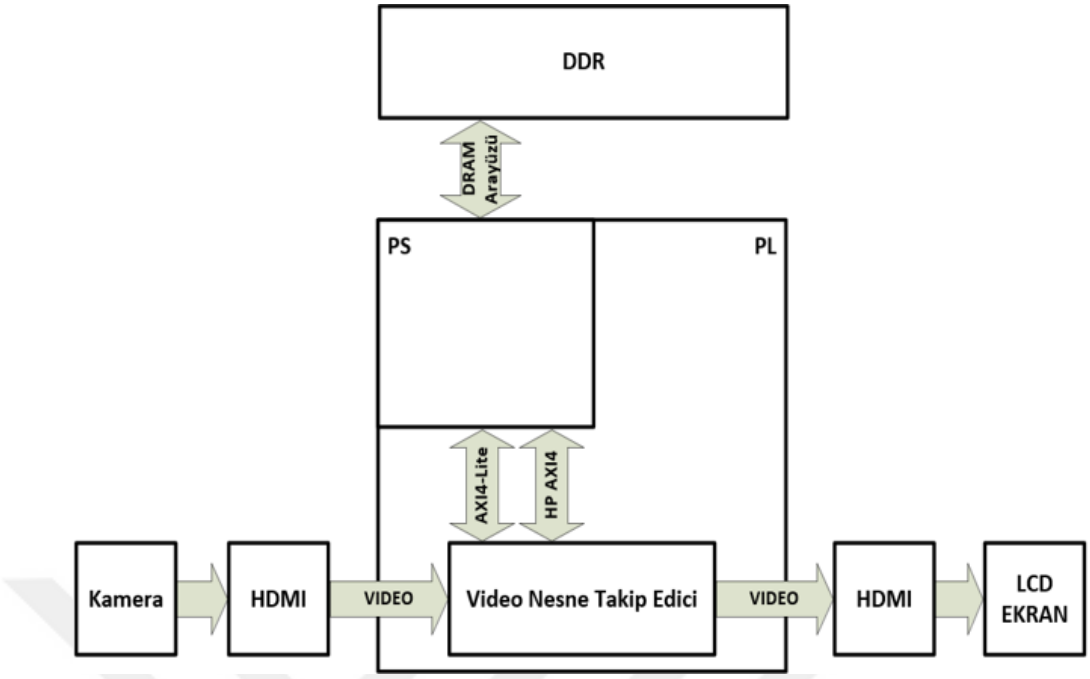
Tez çalışması kapsamında, FPGA donanımı tasarımının ilk adımı olarak, platform seçimi yapılmıştır. MOSSE video nesne takip algoritması yapısında, yüksek yoğunluklu hesaplama ihtiyacı barındırmasının yanında filtre ilklendirilmesi, tasarlanan donanımın durum makinası kontrolü ve haberleşme gereksinimlerini de barındırmaktadır. Bu gereksinimler dikkate alındığında, Xilinx firmasının Zynq SoC platformu tercih edilmiştir. Zynq SoC platformu, tek entegre devre içerisinde FPGA (PL) ve ARM Cortex A-9 (PS) barındırmaktadır. Ek olarak Xilinx firmasının Vivado HLS geliştirme ortamı, yüksek seviye sentez yaklaşımı ile FPGA donanım tasarım imkanı sağlamaktadır. Vivado HLS geliştirme ortamı, FFT dönüşümü, video renk uzayı dönüşümü, trigonometrik hesaplama gibi video işleme ve sinyal işleme donanım kütüphanelerini barındırmaktadır. Tasarlanan sistemde, yüksek boyuttaki verilen işlenmesi ve yüksek hız gereksinimim olduğu kısımlar PL tarafında, sistem kontrolü ve haberleşme gereksinimlerinin olduğu kısımlar ise PS tarafında gerçeklenmesi tercih edilmiştir.

FPGA donanım tasarımı için ise Vivado HLS ve Vivado geliştirme ortamları tercih edilmiştir. Xilinx Zynq-7000 SoC mimarisine Şekil 4.1'de yer verilmiştir. Xilinx Zynq-7000 SoC mimarisinde de görülebileceği üzerine PS ve PL tarafları ayrılmaktadır. PS ve PS tarafları arasındaki haberleşme, AXI arayüzü ile sağlanabilmektedir. PS kısmında ise DDR hafıza 'ya erişebilmek için DRAM kontrol arayüzü bulunmaktadır.



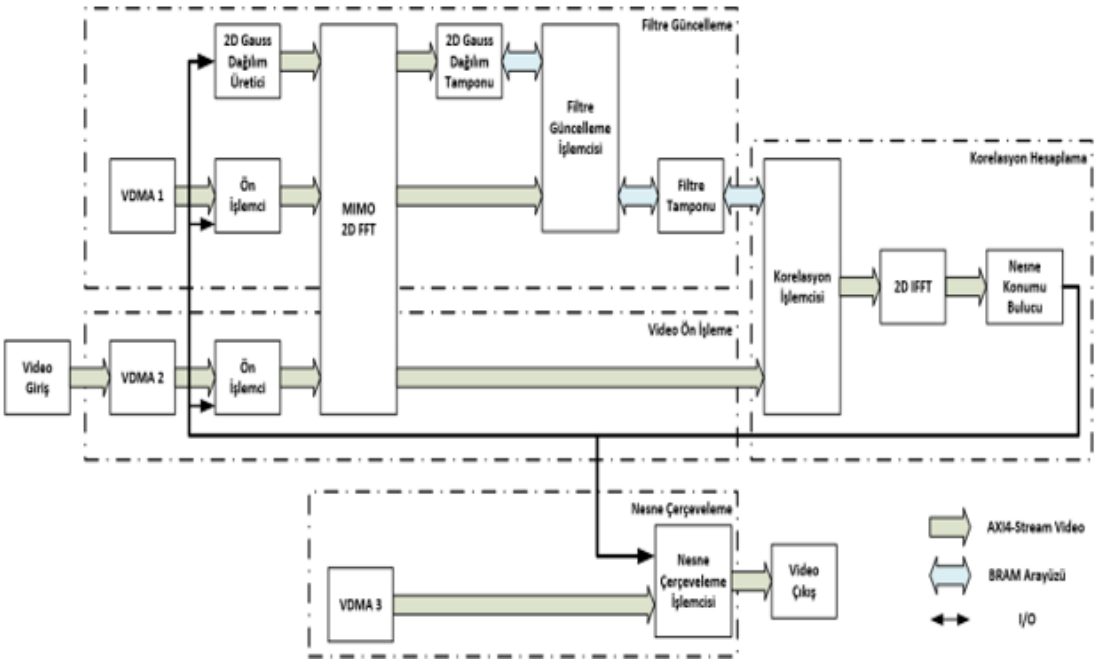
Şekil 4.1. Xilinx Zynq-7000 SoC mimarisi [22]

Video nesne takip edici üst sistem mimarisi, Şekil 4.2'de verilmiştir. Video nesne takip edici sisteme, HDMI arayüzü ile video girişi yapılmıştır. Seçilen nesnenin takip edilmesi ve çerçeve içerisine alınarak işaretlenmesi ile işlenen video, HDMI arayüzü ile sistem dışına çıkış olarak verilmiştir.



Şekil 4.2. Video nesne takip edici üst sistem mimarisi

MOSSE video nesne takip algoritması, FPGA donanım mimarisi Şekil 2.7'deki algoritma blok şemasından yola çıkılarak Şekil 4.3'deki gibi tasarlanmıştır. Tasarlanan FPGA donanım mimarisi, 4 aşamayı gerçekleştirecek şekilde çalışmaktadır.



Şekil 4.3. MOSSE video nesne takip algoritması FPGA donanım mimarisi

- Video ön işlem: Zaman boyutundaki görüntü arama penceresine, önişlemin yapılması ve 2DFFT dönüşümü ile frekans boyutuna geçilmesi aşamasıdır.

- Korelasyon hesaplama: Korelasyon filtresi ve frekans uzayındaki arama penceresi matrisi kullanılarak, korelasyon matrisi hesabının yapılması ve yeni hesaplanan korelasyon matrisi ile yeni nesne konumunun bulunması aşamasıdır.

- Nesne çerçeveleme: Yeni nesne konumu kullanılarak, video çerçevesi üzerine nesne alanının işaretlenmesi aşamadır.

- Filtre güncelleme: Bir sonraki takip adımında kullanılmak üzere korelasyon filtresinin güncellenmesi aşamasıdır.

Tasarlanan donanım mimarisinin temel amacı, sisteme HDMI portu ile girilen video içerisindeki kullanıcı tarafından seçilen nesneyi takip etmek ve nesnenin konumunu, piksel koordinat düzleminde çıkış olarak vermektir. Tasarlanan donanım, takip edilen nesnenin piksel koordinat düzlemindeki konumuna göre video içerisindeki nesneyi çerçevelemektedir ve HDMI portu ile çıkış olarak vermektedir.

Tasarlanan donanım mimarisinde, MOSSE algoritmasının her aşamasında bulunan işlemleri gerçekleştirmek için alt donanım modülleri bulunmaktadır. Bu modüller, ardı ardına bağlanabilecek şekilde, giriş çıkış arayüzlerine sahiptir.

4.1.1. FPGA alt modülleri giriş çıkış arayüzleri

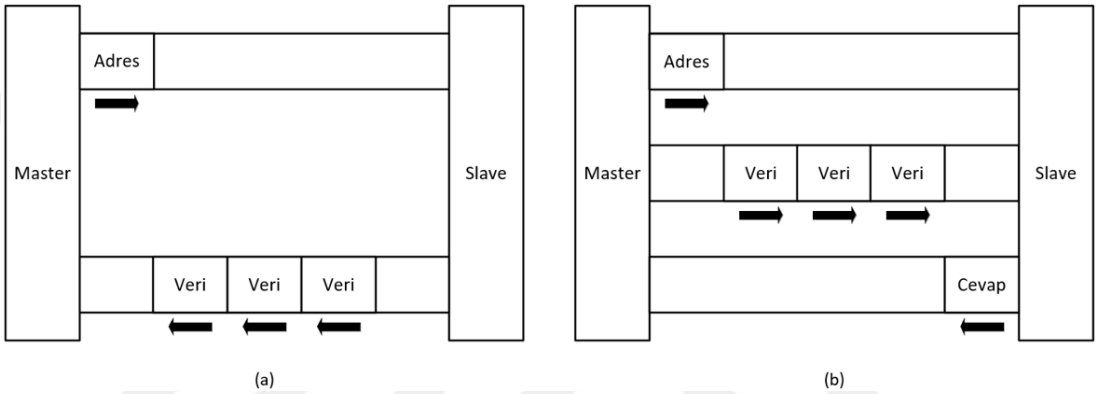
Şekil 4.2'deki MOSSE algoritması üst sistem mimarisinde, video nesne takip edici donanımında kullanılan donanım arayüzleri gösterilmiştir. PS ve PL tarafları arasındaki veri akışı için AXI4-Lite arayüzü kullanılmıştır. Video nesne takip edici donanımına, video girişi ve video çıkışı için HDMI arayüzü kullanılmıştır. PL tarafında tasarlanan FPGA donanımı IP'leri arasında AXI4-Stream Video arayüzü ve BRAM arayüzleri kullanılmıştır.

Tasarımdaki arayüz çeşitliliğinin azaltılması için çıkışı başka bir FPGA donanımı IP'sinin girişine bağlanan FPGA donanımı IP'lerinin, giriş çıkış arayüzlerinin benzer olmasına ve arayüz üzerindeki veri yolunun uzunluğunun aynı olmasına dikkat edilmiştir.

4.1.1.1. AXI4-Lite arayüzü

AXI, ARM AMBA standart ailesine ait entegre içi haberleşme veri yoludur. AXI4, AMBA 4.0 versiyonu ile beraber gelmiştir. AXI4 arayüzünün üç tipi bulunmaktadır. Bunlar AXI4, AXI4-Lite ve AXI4-Stream dir.

AXI4-Lite basit, düşük verimli, bellek erişimi için kullanılan entegre içi haberleşme veri yoludur. Donanım ve çevre birimlerinin kontrolü ve kaydedicilerine erişmek için kullanılmaktadır.



Şekil 4.4. (a) AXI4-Lite Veri Okuma, (b) AXI4-Lite Veri Yazma

Vivado HLS geliştirme ortamı AXI4-Lite veri yolunu desteklemektedir. Zynq SoC mimarisinde, PL tarafında AXI4-Lite veri yoluna sahip bir FPGA donanımı bir temel adres almaktadır. PS tarafına AXI4-Lite veri yolu ile bağlanan PL donanımı bir çevre birimi gibi çalıştırılabilmektedir ve aldığı temel adres üzerinden kayıt edicilerine AXI4-Lite veri yolu ile erişilebilmektedir.

4.1.1.2. AXI4-Stream Video arayüzü

AXI4-Stream Video arayüzü, AXI4-Stream arayüzünü temel alan, video çerçevelerini donanımlar arası iletmek için özelleştirilmiş, ARM AMBA ailesine ait bir entegre içi donanım arayüzüdür. AXI4-Stream Video bir çok yönden AXI4-Stream arayüzü özelliklerini barındırmaktadır. AXI4-Stream Video arayüzü tek yönlü olarak çalışmaktadır. Aktarılacak veri master donanım tarafından slave donanıma iletilmektedir. Slave ve Master donanımların AXI4-Stream Video arayüzü sinyalleri Tablo 4.1 ve Tablo 4.2'de yer almaktadır.

Tablo 4.1. Slave donanım AXI4-Stream Video arayüzü sinyalleri

Sinyal İsmi	Veri Uzunluğu	Veri Yönü	AXI4-Stream Sinyal İsmi
ACLK	1 bit	Giriş	s_axis_video_aclk
Video Data	Aktarılacak veri uzunluğuna göre değişmektedir	Giriş	s_axis_video_tdata
Valid	1 bit	Giriş	s_axis_video_tvalid
Ready	1 bit	Çıkış	s_axis_video_tready
Start of frame	1 bit	Giriş	s_axis_video_tuser
End of frame	1 bit	Giriş	s_axis_video_tlast

Tablo 4.2. Master donanım AXI4-Stream Video arayüzü sinyalleri

Sinyal İsmi	Veri Uzunluğu	Veri Yönü	AXI4-Stream Sinyal İsmi
ACLK	1 bit	Giriş	m_axis_video_aclk
Video Data	Aktarılacak veri uzunluğuna göre değişmektedir	Çıkış	m_axis_video_tdata
Valid	1	Giriş	m_axis_video_tvalid
Ready	1	Çıkış	m_axis_video_tready
Start of frame	1	Çıkış	m_axis_video_tuser
End of frame	1	Çıkış	m_axis_video_tlast

AXI4-Stream arayüzünde her saat sinyalinde, TDATA portuna gönderilecek veri çıkış olarak verilir. Gönderim, master donanımın TVALID portunu yükselen kenar yapması ve ardından slave cihazın TREADY portunu yükselen kenar yapması ile başlamaktadır. Master cihazın TLAST portunu yükselen kenar yapması ile birlikte TDATA portundaki verinin son veri olduğu ve gönderimin bittiği belirtilir. AXI4-Stream arayüzü bir dizi verisinin bir donanımdan başka bir donanıma gönderilmesi için en uygun arayüzlerden biridir.

AXI4-Stream Video arayüzünde AXI4-Stream arayüzünden farklı olarak satır sonu (EOL), çerçeve başlangıcı (SOF) sinyalleri bulunmaktadır. TREADY sinyalinin yükselen kenarı ile birlikle ilk veri TDATA portuna gönderildiğinde, TUSER sinyali yükselen kenar olmaktadır ve verinin video çerçevesinin ilk pikseli olduğunu belirtmektedir. Video çerçevesinin her satırının son pikseli, veri yoluna gönderildiğinde TLAST sinyali yükselen kenar olmaktadır ve video çerçevesinin ilgili satırının son pikselinin, veri yolunda olduğunu belirtmektedir. Bu özellikleri ile AXI4-

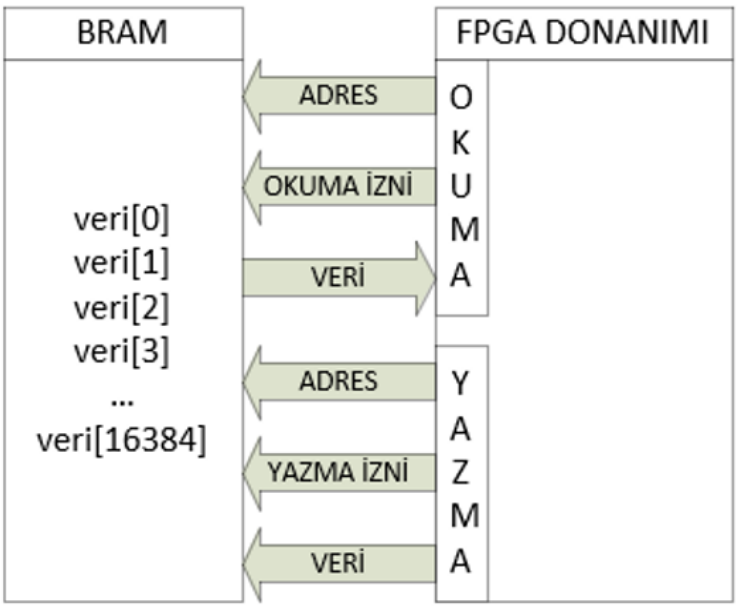
Stream Video arayüzü video çerçevesi veya matris cinsinden verilen donanımlar arası gönderimi için özelleşmiş entegre içi donanım arayüzleridir [23].

Şekil 4.3'deki MOSSE video nesne algoritması donanım mimarisinde bulunan alt modüller arasındaki veri gönderimi için AXI4-Stream Video arayüzü kullanılmıştır. Bölüm 2'de matematiksel modeli açıklanan MOSSE algoritmasında matris işlemleri yoğun olarak kullanılmaktadır. Bu bağlamda bir donanımda üzerinde işlem yapılan bir matrisin, başka bir donanıma gönderimi için AXI4-Stream Video arayüzü kullanılmaktadır.

4.1.1.3.BRAM arayüzü

BRAM arayüzü, RAM ve ROM hafıza elemanlarına erişmek üzere kullanılan entegre içi donanım arayüzüdür. Standart olarak BRAM arayüzünde veri, yazma izni, okuma izni ve adres portları bulunmaktadır. Kullanılan platforma göre hafıza elemanlarının bir veya iki adet BRAM arayüzü bulunmaktadır.

Şekil 4.2'deki donanım mimarisinde, güncellenen filtreler bir sonraki takip adımında kullanılmak üzere BRAM hafıza elemanı kayıt edilmektedir. BRAM hafıza elemanında, dizi formatına çevrilmiş korelasyon filtresi matrisi kayıt edilmektedir ve BRAM arayüzü ile erişilmektedir.

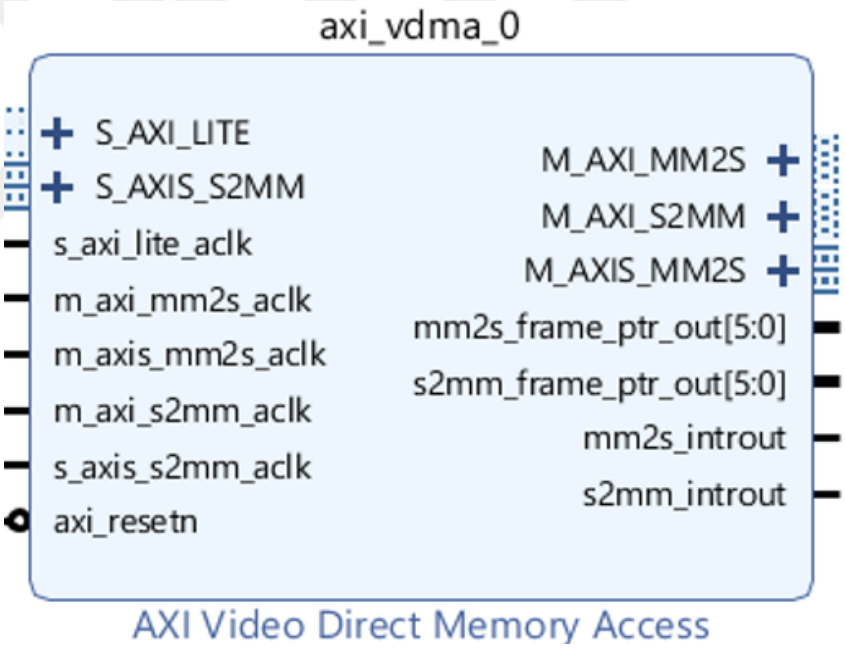


Şekil 4.5. BRAM arayüzü

4.1.2. Video doğrudan bellek erişimi (VDMA)

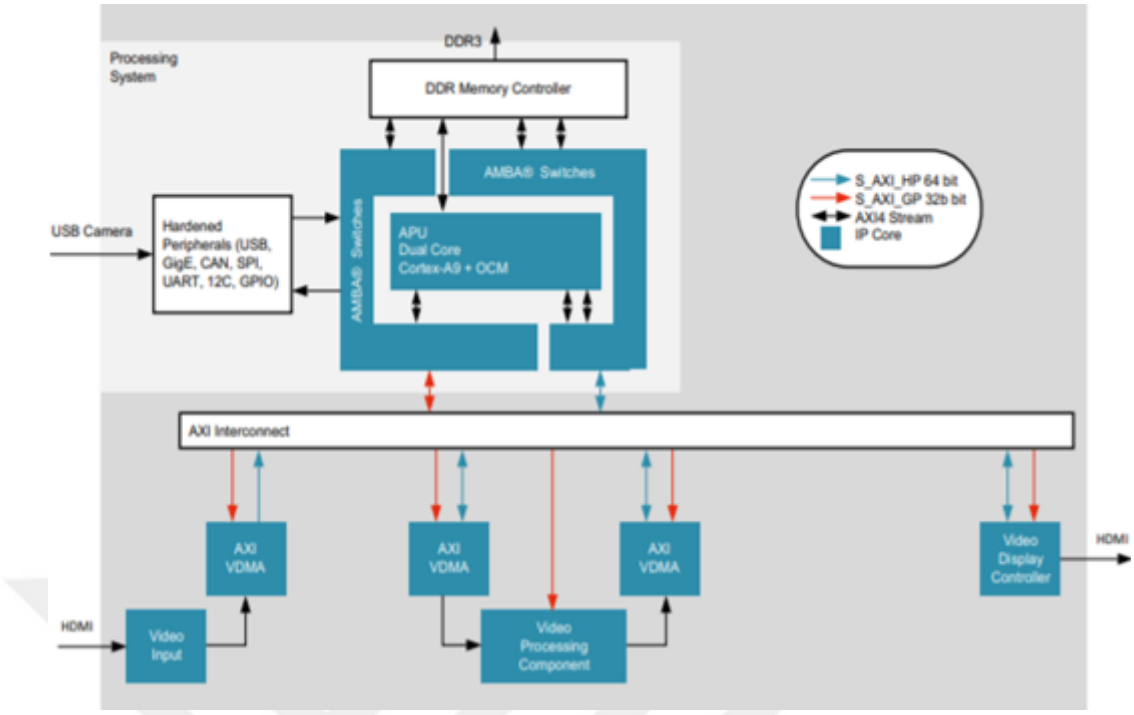
Gömülü sistem donanımlarında, bir bellek ‘ten başka bir belleğe veya bir veri yoluna veri akışı sağlayan ve bu akışı sağlarken başka bir işlem birimine ihtiyaç duymayan çevre birimlerine doğrudan bellek erişimi (DMA) ismi verilmektedir. Video doğrudan bellek erişimi (VDMA) ise görüntü işleme için özelleştirilmiş, bir bellekten veya bir veri yolundan aldığı görüntü çerçevesini başka bir belleğe veya veri yoluna aktaran ve bu işlemi yaparken başka bir işlem birimine ihtiyaç duymayan çevre birimidir.

Görüntü işleme uygulamalarında, saniye başına görüntü çerçevesi değişimi, çözünürlük değişimi veya görüntü çerçevesinden pencere çıkarılması işlemlerinde görüntü tamponuna ihtiyaç duyulmaktadır. Görüntü çerçevesini yüksek hızlı, görüntü tamponuna yazılması ve erişilmesi işlemlerinde VDMA kullanılmaktadır.



Şekil 4.6. Xilinx Vivado VDMA IP gösterimi

Görüntü işleme uygulamalarında görüntü tamponu, görüntü çerçevesi ile aynı boyuttadır. Bir görüntü tamponunun FPGA donanımı olarak tasarlanması, hafıza elemanının boyutunun yüksek olması nedeniyle, maliyetli ve kaynak tüketimi fazla olan bir yöntemdir. Bu duruma çözüm olarak, FPGA donanımının dışardan bir hafıza elemanına erişmesi ve bu hafıza elemanını görüntü tamponu olarak kullanması literatürde tercih edilen bir yöntemdir.



Şekil 4.7. Xilinx Zynq SoC, görüntü tamponu mimarisi [24]

Tez çalışması kapmasında kullanılan Xilinx Zynq SoC mimarisi PS tarafında, AMBA veri yoluna bağlı bir DDR hafıza kontrolcüsüne sahiptir. PL tarafında gerçeklenen sayısal devre, VDMA vasıtasıyla, S_AXI_HP portu üzerinden DDR kontrolcüsüne erişerek DDR belleği görüntü tamponu olarak kullanabilmektedir. Şekil 4.7'de Zynq SoC üzerinde çalışan ve VDMA barındıran bir video işleme mimarisi gösterilmiştir.

Şekil 4.6'de Xilinx Vivado geliştirme ortamının desteklediği bir VDMA IP'si gösterilmektedir. VDMA IP'si okuma, yazma ve hem okuma hem yazma olarak yapılandırılabilmektedir. Yapılandırılma ayarlarına göre giriş çıkış portları değişiklik göstermektedir.

VDMA IP'si üzerindeki portların aşağıdaki gibidir;

- S_AXI_LITE: VDMA IP'sinin kontrolünün sağlanması amacıyla kullanılan AXI4 portudur. VDMA IP'si bu port üzerinden Master donanıma bağlanmaktadır. VDMA IP'sinin kaydedici yapılandırmaları bu port üzerinden yapılmaktadır.

- S_AXIS_S2MM: AXI4-Stream Video arayüzü olarak çalışan giriş portudur. Bu port ile VDMA'e giriş yapan video çerçevesi, VDMA'in yapılandırılan DDR adreslerine göre DDR hafızaya üzerine yazılmaktadır.

- M_AXIS_MM2S: AXI4-Stream Video arayüzü olarak çalışan çıkış portudur. Bu port ile VDMA'in yapılandırılan DDR adreslerine göre video çerçevesi, DDR hafızadan okunmakta ve çıkış olarak verilmektedir.

- M_AXI_S2MM: VDMA'in DDR hafızaya görüntü çerçevesini yazmak için AXI Interconnect ile PS tarafındaki AMBA veri yoluna erişmesini sağlayan porttur.

- M_AXI_MM2S: VDMA'in DDR hafızaya yazılan görüntü çerçevesini erişmek için AXI Interconnect ile PS tarafındaki AMBA veri yoluna erişmesini sağlayan porttur.

Video nesne takip işleminin t zamanındaki video çerçevesi, arama penceresi çıkarımında, filtre güncelleme işleminde ve nesne çerçeveleme işleminde kullanılmaktadır. Bu doğrultuda t zamanındaki video çerçevesinin bir hafıza elemanına kayıt edilmesi gerekmektedir. Kaynak tüketimi dikkate alındığında görüntü çerçevesinin kayıt edilmesi için DDR hafıza elemanı kullanılmaktadır. Şekil 4.2'deki MOSSE algoritması donanım mimarisinde üç adet VDMA kullanılmaktadır.

4.1.2.1. VDMA yazma portunun yapılandırılması

Xilinx VDMA IP'sinin yazma portunu yapılandırılması için Tablo 4.3'deki kaydediciler kullanılmaktadır.

Tablo 4.3. Xilinx VDMA IP'si S2MM portu yapılandırma kaydedicileri

Kaydedici ismi	Kaydedici Adresi	Kaydedici açıklaması
S2MM_VDMACR	0x30 - 0x8B	VDMA kontrol kaydedici
S2MM_START_Address	0xAC - 0xB4	VDMA yazma başlangıç adresi
S2MM_FRMDLY_STRIDE	0xA8	Çerçevenin satırlarının kaç byte'ının yazılacağını ifade eder
S2MM_HSIZE	0xA4	Çerçeve satır başına bayt sayısı
S2MM_VSIZE	0xA0	Çerçevede bulunan satır sayısı

- S2MM_START_Address kaydedicisine, video çerçevesinin DDR hafızada hangi adresten itibaren yazılacağını ifade tanımlanmaktadır.

- S2MM_HSIZE kaydedicisi, DDR hafızaya yazılacak görüntü çerçevesinin bir satırının kaç bayt uzunluğunun yazılacağını ifade etmektedir. 640 × 480 RGB bir

video çerçevesinin bir satırında 640 piksel bulunmaktadır. Her renk bileşeni 1 byte uzunluğundadır ve her piksel 3 renk bileşeninden oluşmaktadır. Bu değerler hesaba katıldığında 640×480 RGB bir video çerçevesinin her satırında 1920 byte uzunluğunda veri bulunmaktadır. Bu doğrultuda S2MM_HIZE kaydedicisine 1920 değeri yüklenmektedir.

- S2MM_VSIZE kaydedicisi, DDR hafızaya yazılacak video çerçevesinin kaç satırdan oluştuğunu ifade etmektedir. 640 × 480 RGB bir video çerçevesinde 480 tane satır bulunmaktadır. Bu doğrultuda S2MM_VSIZE kaydedicisine 480 değeri yüklenmektedir.

- S2MM_FRMDLY_STRIDE kaydedicisi, DDR hafızaya yazılacak olan görüntü çerçevesinin bir satırının kaç bayt uzunluğundaki veriden oluştuğunu ifade etmektedir. 640 × 480 boyutlarındaki RGB görüntü çerçevesinin bir satırı 1920 bayt uzunluğundaki veriden oluşmaktadır. Bu doğrultuda S2MM_FRMDLY_STRIDE kaydedicisine 1920 değeri yüklenmektedir.

- S2MM_VDMACR kaydedicisi ile VDMA yapılandırılan ayarlarda çalıştırılmaktadır.

4.1.2.2. VDMA okuma portunun yapılandırılması

Xilinx VDMA IP'sinin yazma portunu yapılandırılması için Tablo 4.4'deki kaydediciler kullanılmaktadır.

Tablo 4.4.Xilinx VDMA IP'si MM2S portu yapılandırma kaydedicileri

Kaydedici ismi	Kaydedici Adresi	Kaydedici açıklaması
MM2S_VDMACR	0x30 - 0x8B	VDMA kontrol kaydedici
MM2S_START_Address	0x5C - 0x64	VDMA okuma başlangıç adresi
MM2S_FRMDLY_STRIDE	0x58	Çerçevenin satırlarından okunacak bayt sayısı
MM2S_HSIZE	0x54	Çerçeve satır başına bayt sayısı
MM2S_VSIZE	0x50	Çerçevede bulunan satır sayısı

- MM2S_START_Address kaydedicisine, video çerçevesinin DDR hafızadan hangi adresten itibaren okunacağını ifade etmektedir.

- MM2S_HSIZE kaydedicisi, DDR hafızadan okunacak görüntü çerçevesinin bir satırının kaç baytının okunacağını ifade etmektedir. 640×480 RGB bir video çerçevesinin bir satırında 640 piksel bulunmaktadır. Her renk bileşeni 1byte'tır ve her piksel 'de 3 byte veri bulunmaktadır. 640×480 RGB bir video çerçevesinin her satırında 1920 byte veri bulunmaktadır. Bu bağlamda MM2S_HIZE kaydedicisine 1920 değeri yüklenmiştir.

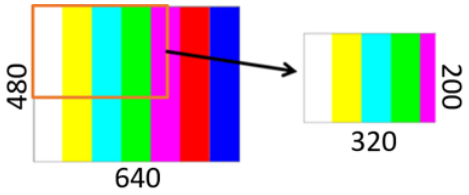
- MM2S_VSIZE kaydedicisi, DDR hafızadan okunacak video çerçevesinin kaç adet satırdan oluştuğunu ifade etmektedir. 640×480 RGB bir video çerçevesinde 480 tane satır bulunmaktadır. Bu bağlamda MM2S_VSIZE kaydedicisine 480 değeri yüklenmiştir.

- MM2S_FRMDLY_STRIDE kaydedicisi, DDR hafızadan okunacak olan görüntü çerçevesinin bir satırının kaç bayt veriden oluştuğunun yazıldığı kaydedicidir. 640×480 RGB bir görüntü çerçevesinin bir satırında 1920 bayt veri bulunmaktadır. Bu nedenle MM2S_FRMDLY_STRIDE kaydedicisine 1920 değeri yüklenmiştir.

- MM2S_VDMACR kaydedicisi ile VDMA yapılandırılan ayarlarda çalıştırılmaktadır.

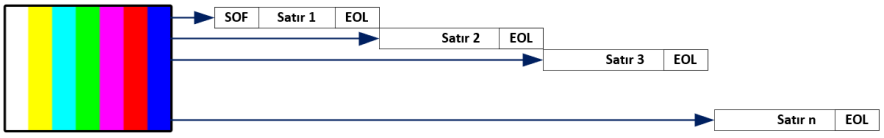
4.1.2.3. VDMA ile arama penceresi çıkarımı

Bölüm 2'de arama penceresi çıkarımı açıklanmıştır. Video çerçevesi üzerinden bir arama penceresi çıkarımı için video çerçevesinin bir görüntü tamponuna depolanması gerekmektedir. Görüntü tamponunun FPGA donanımı olarak gerçeklenmesi, maliyetli ve yüksek alan kaplayan bir tercih olmaktadır. Bu nedenle arama penceresi çıkarımı için gerekli olan görüntü tamponunun DDR hafızada tutulması tercih edilmiştir.



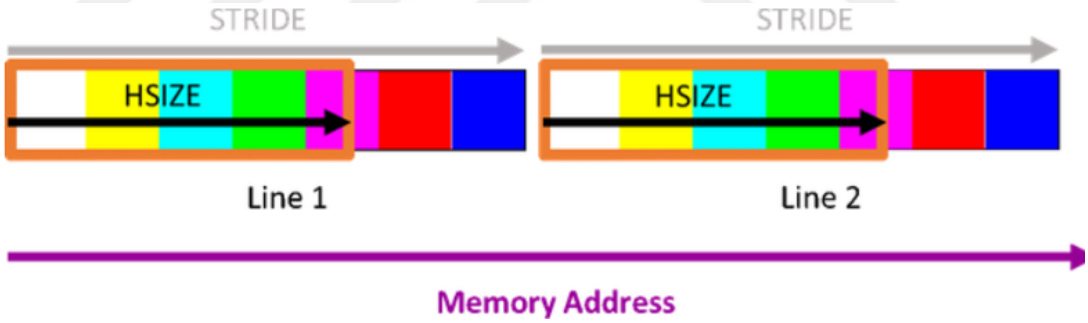
Şekil 4.8. Arama penceresi çıkarımı gösterimi[25]

Xilinx VDMA IP'si DDR hafızadaki görüntü tamponundan arama çerçevesi çıkarımında kullanılmak üzere yapılandırılabilmektedir. Bu doğrultuda Şekil 4.2'deki MOSSE algoritması donanım mimarisinde video ön işlem adımında, video çerçevesini DDR hafızaya yazılması ve DDR hafızaya yazılan video çerçevesinden arama penceresi çıkarımı için VDMA kullanılmıştır. Kullanılan VDMA'in yazma ve okuma portları kullanılmıştır. Arama penceresi çıkarımı için VDMA'in okuma portu kullanılmıştır.



Şekil 4.9. Video çerçevesinin AXI4- Stream Video ile gönderimi

Tamponlanmış görüntü çerçevesinden arama penceresi çıkarımı işlemi, temel olarak görüntü çerçevesinin her satırından, arama penceresinin bir satırının uzunluğu kadar veri okunması işlemidir. Xilinx VDMA IP çekirdeği bu şekilde olan görüntü çerçevesi okuma işlemini desteklemektedir.

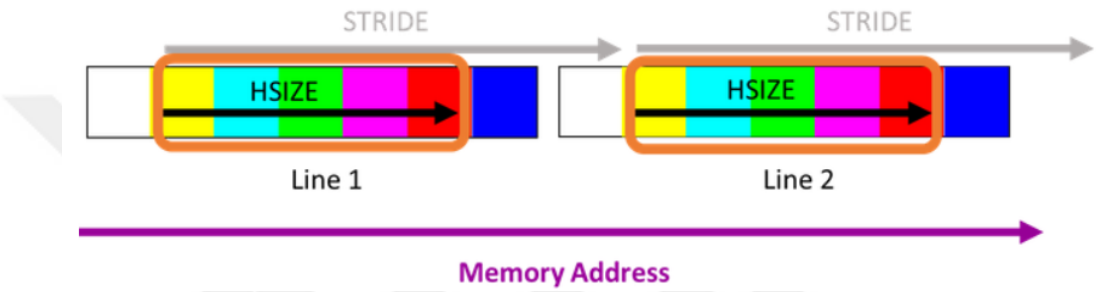


Şekil 4.10. Görüntü çerçevesinden arama penceresi satırının okunması [25]

VDMA'in tamponlanmış görüntü çerçevesinden arama penceresi çıkarımı yaparken, Şekil 4.10'da görüleceği üzere DDR hafızanın başlangıç adresinden arama penceresi satır boyutu kadar veri okuması ve bir sonraki satıra geçmesi için arta kalan satır veri boyutu kadar olan DDR adresini atlaması gerekmektedir. VDMA'in bu adres atlaması yapacak şekilde yapılandırılması için MM2S_HSIZE, MM2S_VSIZE ve MM2S_FRMDLY_STRIDE kaydedicileri kullanılmaktadır.

Örnek olarak Şekil 4.8'deki 640×480 boyutlarında bir görüntü çerçevesinden 320×200 boyutunda arama penceresi çıkarımını ele alındığında, görüntü çerçevesinin bir

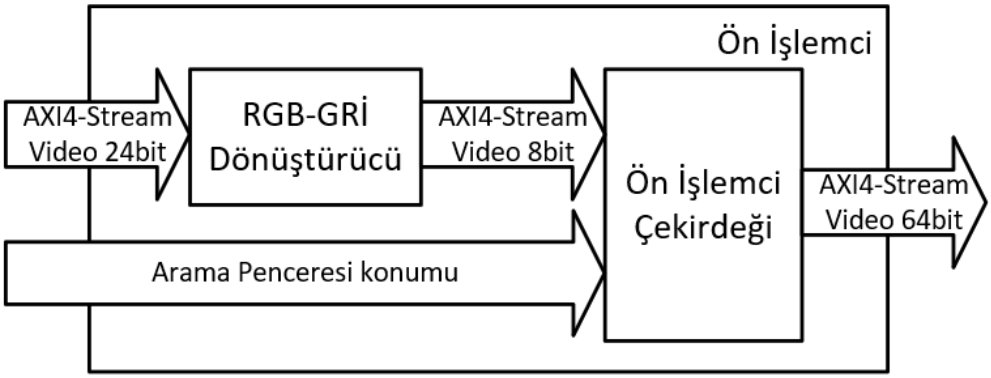
satırında, 1920 bayt veri olması nedeniyle MM2S_FRMDLY_STRIDE kaydedicisine 1920 değeri yüklenmektedir. Arama penceresinin bir satırında 320 piksel olduğundan ve her piksel değerinin 3 bayt veri barındırmasından dolayı, arama penceresinin bir satırında 960 bayt veri bulunmaktadır. Bu nedenle MM2S_HSIZE kaydedicisine 960 değeri yüklenmektedir. Arama penceresi 200 satırdan oluştuğundan dolayı MM2S_VSIZE kaydedicisine 200 değeri yüklenmektedir. Arama penceresinin başlangıç konumu görüntü çerçevesi üzerinde değiştiği durum ele alındığında ise MM2S_START_Address kaydedicisine başlangıç adresi yüklenmektedir.



Şekil 4.11. Arama penceresinin görüntü çerçevesi hareketi [25]

4.1.3. Ön İşlemci donanım tasarımı

Şekil 4.2'deki MOSSE algoritması FPGA donanım mimarisinde, filtre güncelleme ve video ön işlem adımlarında, görüntü ön işlemci donanımı kullanılmıştır. Ön işlemci donanımı giriş arayüzünden giriş yapılan RGB renk uzayındaki görüntüyü ilk olarak gri renk uzayına dönüştürmektedir. Gri renk uzayındaki görüntü üzerinde denklem (2.11)'deki normalizasyon işlemini yapılmakta ve normalize edilen görüntü bir iki boyutlu çerçeve fonksiyonu matrisi ile çarpılmaktadır.



Şekil 4.12. Ön İşlemci donanım mimarisi

Ön işlemci donanımı üst seviye donanım mimarisi Şekil 4.12'de gösterilmiştir. Ön işlemci donanımı, RGB-GRİ dönüştürücü ve ön işlemci çekirdeği donanımlarından oluşmaktadır. Ön işlemci donanımı giriş çıkış arayüzleri Tablo 4.5'de verilmiştir.

Tablo 4.5. Ön işlemci donanımı giriş çıkış arayüzleri

Arayüz İsmi	Arayüz Türü	Giriş/Çıkış Türü	Veri Boyutu
Arama Penceresi Girişi	AXI4-Stream Video	Giriş	24bit
Arama Penceresi Konumu	I/O	Giriş	32bit
Arama Penceresi Çıkışı	AXI4-Stream Video	Çıkış	64bit

Arama penceresi girişi arayüzü ile ön işlemci donanımına, işlenecek video girişi yapılmıştır. Arama penceresi arayüzü tdata sinyali veri içeriği Tablo 4.6'de verilmiştir.

Tablo 4.6. Arama penceresi giriş arayüzü veri içeriği

Sinyal/Alan Adı	Veri Tipi	Byte	Açıklama
Kırmızı Bileşen	uint8	1	Piksel kırmızı bileşeni
Yeşil Bileşen	uint8	1	Piksel yeşil Bileşeni
Mavi Bileşen	uint8	1	Piksel mavi bileşeni

Arama penceresi konumu giriş arayüzü arama penceresinin, görüntü çerçevesi üzerinde piksel koordinat düzleminde konumunu ve boyutlarını içermektedir. Arama penceresi konumu giriş arayüzü veri içeriği Tablo 4.7'de verilmiştir.

Tablo 4.7. Arama penceresi konumu giriş arayüzü veri içeriği

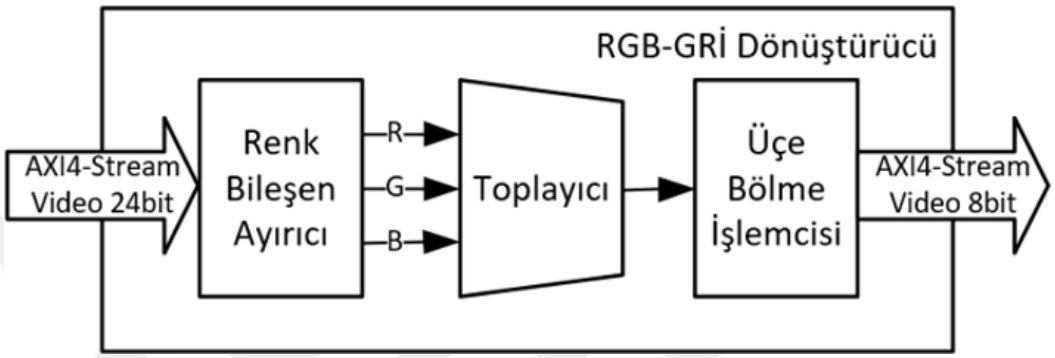
Sinyal/Alan Adı	Veri Tipi	Byte	Açıklama
x	uint8	1	Arama penceresi x konumu
y	uint8	1	Arama penceresi y konumu
w	uint8	1	Arama penceresi genişliği
h	uint8	1	Arama penceresi uzunluğu

Arama penceresi çıkışı arayüzü, arama penceresinin ön işlenmiş halde çıkış olarak verildiği arayüzdür. Arama penceresi çıkış arayüzü tdata sinyali veri içeriği Tablo 4.8'de verilmiştir.

Tablo 4.8. Arama penceresi çıkış arayüzü veri içeriği

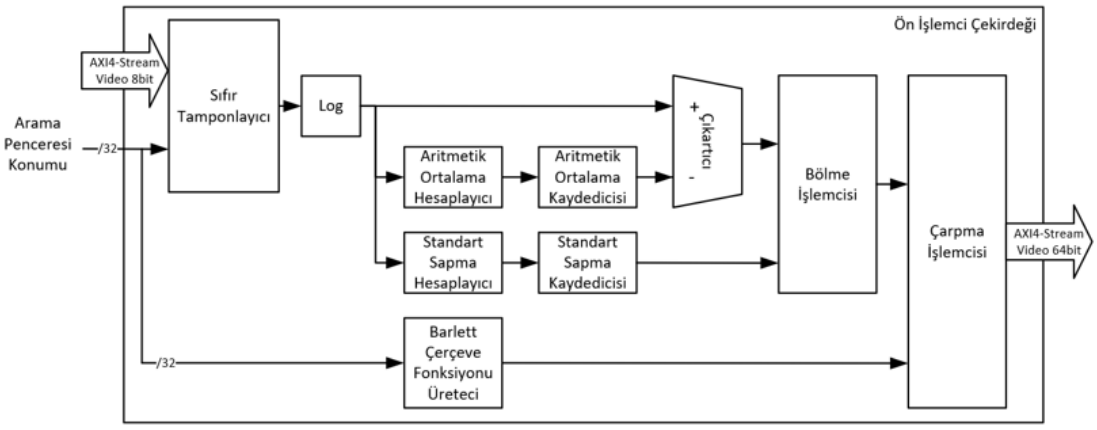
Sinyal/Alan Adı	Veri Tipi	Byte	Açıklama
Gerçek sayı	float	4	Karmaşık sayı gerçek sayı bileşeni
Sanal sayı	float	4	Karmaşık sayı sanal sayı bileşeni

Şekil 4.12’deki ön işlemci donanım mimarisi içerisinde RGB-GRİ dönüştürücü donanımını bulundurmaktadır. RGB-GRİ dönüştürücü, RGB renk uzayında giriş yapılan görüntüyü, gri renk uzayına dönüştürmektedir. Giriş çıkış arayüzü olarak AXI4-Stream Video arayüzü kullanılmıştır. RGB renk uzayından, gri renk uzayına dönüşüm işlemi için RGB renk bileşenlerinin aritmetik ortalaması alınmaktadır. RGB-GRİ dönüştürücü donanım mimarisi Şekil 4.13’de verilmiştir.



Şekil 4.13. RGB-GRİ dönüştürücü donanım mimarisi

Şekil 4.12’deki ön işlemci donanım mimarisinde ön işlemci çekirdeği donanımı bulunmaktadır. Ön işlemci çekirdeği donanımı, RGB renk uzayından gri renk uzayına dönüştürülmüş görüntü üzerinde denklem (2.11)’deki gibi normalizasyon işlemi yapmaktadır. Normalize edilen görüntü 2DFFT dönüşümü sırasında oluşan spektral gürültüleri önlemek amacıyla çerçeve fonksiyonu ile çarpılmaktadır.

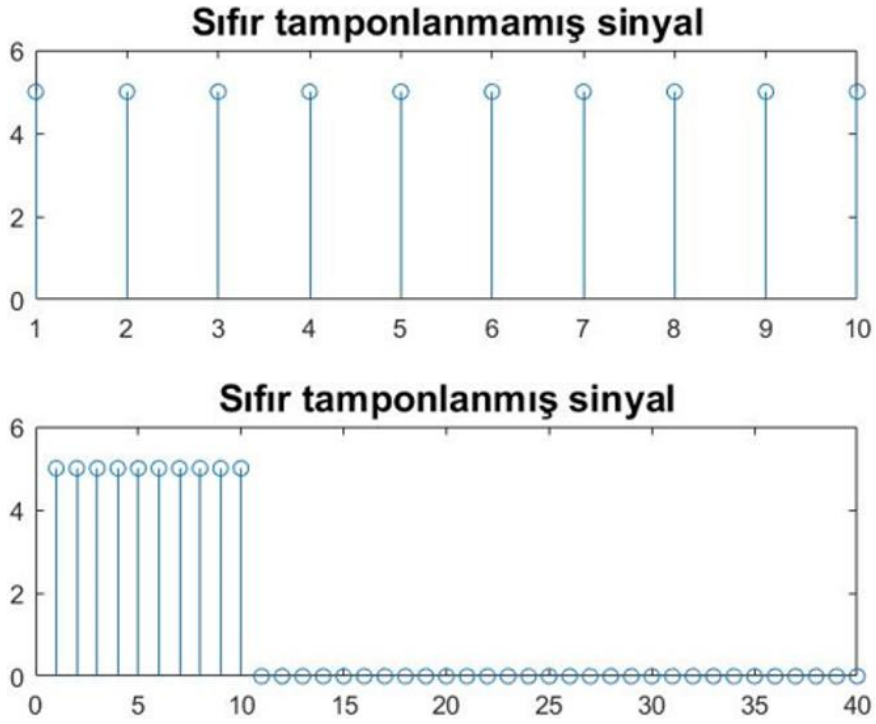


Şekil 4.14. Ön işlemci çekirdeği donanım mimarisi

Ön işlemci çekirdeği donanım mimarisi Şekil 4.14’de verilmiştir. Ön işlemci çekirdeği donanımına, AXI4-Stream Video arayüzü üzerinden gri renk uzayında görüntü matrisi

ve I/O arayüzü ile arama penceresi konumu girişi yapılmaktadır. AXI4-Stream Video arayüzü ile ön işlenmiş görüntü matrisi çıkışı yapılmaktadır.

2DFFT donanımı içerisinde kullanılan ve Vivado HLS'in desteklediği Xilinx FFT IP'si donanımı için sentezlenmeden önce FFT dönüşümü nokta sayısı yapılandırılmaktadır. Bu doğrultuda, bir FFT dönüştürücü donanımı sentezlenmektedir ve nokta sayısı sabit kalmaktadır. Tez çalışması kapsamında 128×128 boyutlarında bir arama penceresi kullanılmaktadır. Takip algoritması ilklendirilmesi aşamasında, 128×128 boyutu büyüklüğüne kadar olan her arama penceresi boyutu seçilebilmektedir. Bu durum FFT dönüşümü işleminde, en fazla 128 noktalı olmak üzere 128 noktaya kadar olan sayılarda FFT dönüşümü yapılabilmesi gereksinimini oluşturmaktadır. Bu gereksinim sabit noktalı FFT dönüşümü yapacak şekilde yapılandırılabilen Xilinx FFT IP'si için bir darboğaz oluşturmaktadır. Literatürde bu duruma çözüm olarak FFT dönüşü yapılacak sinyal'e sıfır tamponlama işlemi yapılmaktadır. Bu nedenle ön işlemci çekirdeği donanım mimarisi, görüntü matrisini normalize etmeden önce sıfır tamponlayıcı donanımı ile görüntü matrisi üzerine iki boyutlu sıfır tamponlama işlemi yapmaktadır.



Şekil 4.15. Bir sinyalin sıfır tamponlanması (n = 10, m = 40)

Sıfır tamponlama, n uzunluğundaki sinyal tamponunun uzunluğunu m uzunluğuna tamamlamak için (m > n) arta kalan tampon elemanlarını sıfır ile doldurulması işlemidir. Şekil 4.15'de bir sinyalin sıfır tamponlanması işlemi gösterilmektedir.

Şekil 4.16. Bir görüntü matrisinin 2 boyutlu sıfır tamponlanması

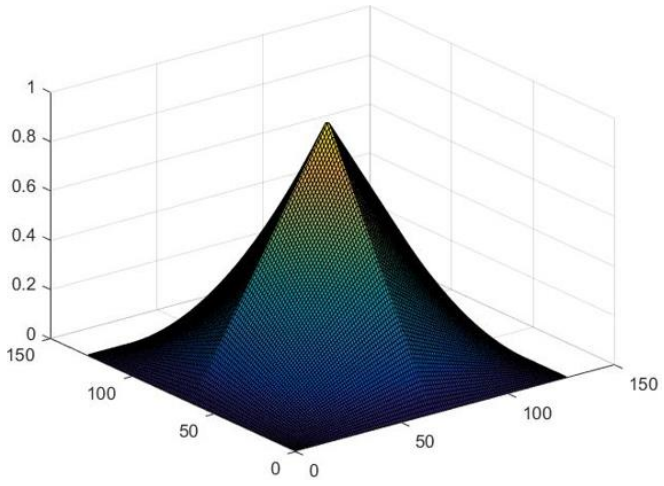
Sıfır tamponlama işleminden sonra normalizasyon işleminin bir parçası olan logaritma alma işlemi yapılmıştır. Logaritma işlemi için Vivado HLS'in desteklediği matematik kütüphanesi kullanılmıştır.

Logaritma işleminin ardından normalizasyon işlemi için denklem (2.11)'deki işlemi gerçekleyen donanım bulunmaktadır. Denklem (2.11)'deki normalizasyon işlemi içerisinde aritmetik ortalama ve standart sapma işlemleri bulunmaktadır. AXI4-Stream Video arayüzü ile gönderilen bir görüntü matrisinin, kendi aritmetik ortalama ve standart sapma değerleri kullanılarak normalize edilmesi için görüntü matrisinin tamponlanarak aritmetik ortalamasının ve standart sapmasının hesaplanması gerekmektedir. Bu durum, ön işlemci çekirdeği donanımında bir görüntü tamponu gereksinimi ortaya çıkarmaktadır. Ön işlemci donanımının görüntü tamponu kullanılarak aritmetik ortalama ve standart sapma değerlerinin hesaplaması kaynak tüketimini ve maliyeti arttırmakta donanım çalışma hızını ise azaltmaktadır.

Görüntü matrisinin aritmetik ortalama ve standart sapma değerlerinin hesaplanmasında, görüntü tamponu kullanımından kaçınmak için ardışık iki görüntü matrisinin ilkinden elde edilen aritmetik ortalama ve standart sapma değerleri ikinci görüntü matrisinin normalizasyon işleminde kullanılmıştır. Gerçek zamanlı (25 fps'den büyük çerçeve hızları için) bir video'daki ardışık iki görüntü matrisinde bulunan ve takip edilen nesne konumunun görüntü matrisleri üzerindeki konumlarına

göre konumlandırılan iki ardışık arama penceresi, birbirlerine çok yakın ve optik akış miktarının çok düşük olduğu istatistiksel olarak kabul edilmektedir. Bu kabul neticesinde ardışık görüntü matrislerinin aritmetik ortalama ve standart sapma değerleri birbirlerine çok yakın değerler olduğu kabul edilmektedir. Bu varsayım neticesinde, görüntü tamponu kullanılmadan aritmetik ortalama ve standart sapma değerleri hesaplanmıştır. Bu tasarım kaynak tüketimi ve hız açısından ön işlemci çekirdeği donanımına kazanç sağlamıştır. Ardışık görüntü matrislerinin ilkinde hesaplanan aritmetik ortalama ve standart sapma değerleri, ikinci görüntü matrisinin normalizasyonunda kullanılmak üzere kaydecilere kayıt edilmektedir. Logaritması alınan görüntü matrisinin, aritmetik ortalama ve standart sapma değerleri bir sonraki görüntü matrisinde kullanılmak üzere aritmetik ortalama hesaplayıcı ve standart sapma hesaplayıcı donanımlarında hesaplanmakta ve görüntü matrisinin son pikselide normalize edilmesiyle kaydedicilere kaydedilmektedir.

Normalizasyon işleminin ardından görüntü matrisi, çerçeve fonksiyonu üretici modülünün ürettiği iki boyutlu çerçeve fonksiyonu matrisi ile eleman bazlı olarak çarpılmaktadır. Çerçeve fonksiyonu üretici donanımı, iki boyutlu barlett çerçeve fonksiyonunu üretmektedir. Barlett çerçeve fonksiyonu içerisinde trigonometrik ifadeler barındırmaması nedeniyle hızlı olarak hesaplanabilmektedir.



Şekil 4.17. 2D Barlett çerçeve fonksiyonu grafiği

Hesaplanan iki boyutlu barlett çerçeve fonksiyonunun ana kolu, arama penceresinin merkeziyle kesişmektedir. İki boyutlu barlett çerçeve fonksiyonu Şekil 4.17'de gösterilmiştir.

$$a(i)=\begin{cases} \frac{2i}{w}, & 0 \le i \le \frac{w}{2} \\ 2-\frac{2i}{w}, & \frac{w}{2} \le i \le w \end{cases} \quad (4.1)$$

$$b(i)=\begin{cases} \frac{2j}{h}, & 0 \le j \le \frac{h}{2} \\ 2-\frac{2j}{h}, & \frac{h}{2} \le j \le h \end{cases} \quad (4.2)$$

Tek boyutlu barlett çerçeve fonksiyonu üretiminde denklem (4.1) ve denklem (4.2) kullanılmıştır. Denklem (4.1) satır ekseni üzerindeki çerçeve fonksiyonu hesaplanmasında denklem (4.2) sütun ekseni üzerindeki çerçeve fonksiyonu hesaplanmasında kullanılmıştır. i ve j değişkenleri iki boyutlu çerçeve fonksiyon matrisinin satır ve sütun eleman numaralarını, w ve h değişkenleri ise arama penceresi boyutlarını ifade etmektedir. İki boyutlu çerçeve fonksiyonu üreteci, tek boyutlu çerçeve fonksiyonu eşitlikleri olan denklem (4.1) ve denklem (4.2) 'den türetilerek denklem (4.3)'deki gibi hesaplanmıştır.

$$w(i,j)=a(i)\times b(j) \quad (4.3)$$

Şekil 4.12'de donanım mimarisi verilen ön işlemci donanımı, Vivado HLS ortamında C++ dili kullanılarak FPGA donanımı olarak tasarlanmıştır. Tasarlanan ön işlemci FPGA donanımı, Vivado HLS ortamında sentezlenmiştir. Ön işlemci donanımı FPGA donanımı sentez sonuçları Tablo 4.9'da yer almaktadır.

Tablo 4.9. Ön İşlemci FPGA donanımı sentez sonuçları

BRAM_18K	DSP48E	FF	LUT	URAM
0	25	4496	6810	0

Vivado HLS ortamında sentezlenen ön işlemci donanımı zamanlama çıktıları, Tablo 4.10'de yer almaktadır.

Tablo 4.10. Ön İşlemci donanımı zamanlama çıktıları

Hedef Saat Frekansı	Kestirilen Saat Frekansı (f_{max})	Belirsizlik	Gerekli Saat Sinyali ($tick_{max}$)
100Mhz	111.35Mhz	1Mhz	82039

$$fps = \frac{1}{f_{max} \times tick_{max}} \quad (4.4)$$

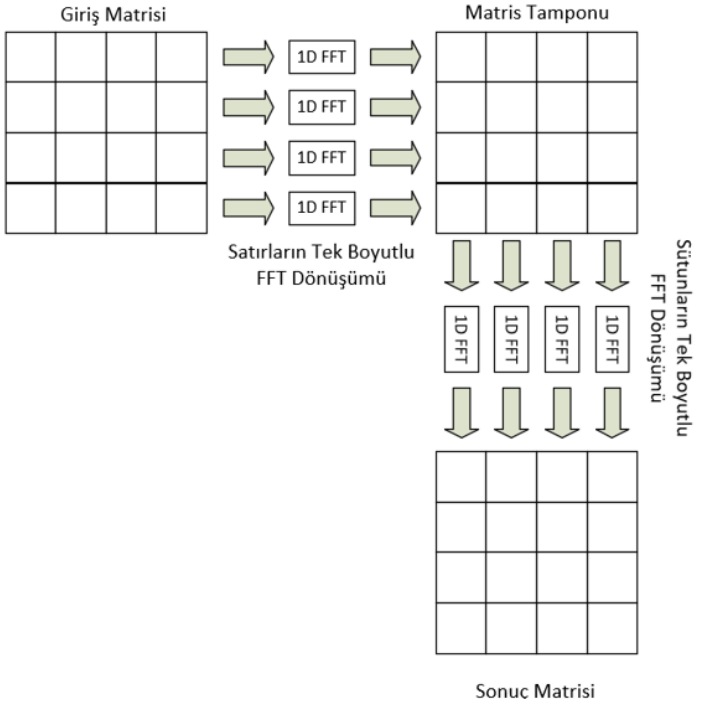
Tasarlanan ön işlemci donanımının, 128 × 128 boyutlarındaki bir arama penceresinin ön işlemi için ulaşabileceği en yüksek fps miktarının hesaplanması için denklem (4.4)'den faydalanılmaktadır.

Ön işlemci donanımı 100Mhz bir saat frekansı ile çalıştırıldığında, 128 × 128 boyutlarındaki bir arama penceresini, @1218.93fps hızında ön işleyebilmektedir.

4.1.4. 2D FFT Dönüştürücü donanım tasarımı

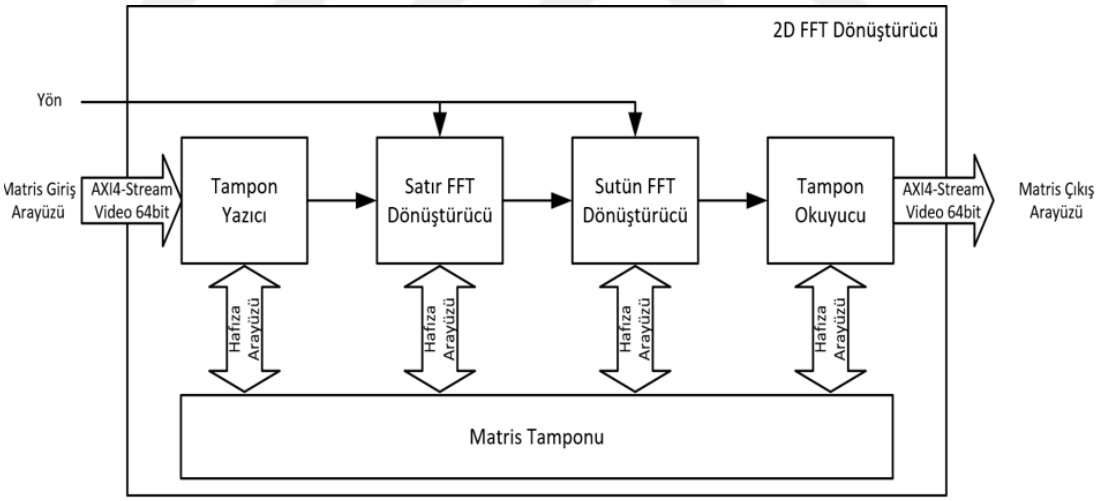
Şekil 4.2'deki MOSSE algoritması FPGA donanım mimarisinde, zaman boyutundaki arama penceresi matrisini, frekans boyutuna taşımak için video ön işlem ve filtre güncelleme adımlarında, frekans boyutundaki korelasyon matrisini ise zaman boyutuna taşımak için korelasyon hesaplama adımında 2D FFT dönüştürücü donanımı kullanılmıştır.

2D FFT dönüşümü, bir matrisin ilk olarak satırlarının tek boyutlu FFT dönüşümünün yapılıp ardından her bir sütununun tek boyutlu FFT dönüşümünün yapılması işlemidir.



Şekil 4.18. Bir matrisin 2D FFT dönüşümü

Şekil 4.18'de bir matrisin 2D FFT dönüşümü işlemi gösterilmiştir. Bir m satır ve n sütundan oluşan matrisin 2D FFT dönüşümü için m×n adet FFT dönüşümü gerçekleştirilmektedir. 2D FFT dönüşümü sırasında, giriş matrisinin satırlarının FFT dönüşümü sonuçlarının, sütunların FFT dönüşümü için kullanılabilmesi için bir matris tamponunda tutulması gerekmektedir. Bu matris tamponunun, arama penceresi boyutlarında olan karmaşık sayı matrisini kaydedebilecek kapasitede olması gerekmektedir. Matris tamponunun boyutlarının büyük olması, kaynak tüketimi ve maliyet açısından olumsuz sonuç oluşturmaktadır. Matris tamponunun küçük seçilmesi ise arama penceresinin küçük seçilmesine sebep olacak ve takip algoritmasının performansına olumsuz yönde etkileyecektir. Bu tez çalışması kapmasında arama penceresi boyutlarının seçiminde, 2D FFT dönüştürücü donanımı matris tamponu boyutları dikkate alınmıştır. 2D FFT dönüştürücü FPGA donanım mimarisi Şekil 4.19'de verilmiştir. 2D FFT dönüştürücü donanımı Tampon Yazıcı, Satır FFT Dönüştürücü, Sütun FFT Dönüştürücü ve Tampon Okuyucu donanımlarından oluşmaktadır.



Şekil 4.19. 2D FFT dönüştürücü FPGA donanım mimarisi

2D FFT dönüştürücü donanımı giriş çıkış arayüzleri Tablo 4.11'de verilmiştir.

Tablo 4.11. 2D FFT dönüştürücü donanımı giriş çıkış arayüzleri

Arayüz İsmi	Arayüz Türü	Giriş/Çıkış Türü	Veri Boyutu
Yön Girişi	I/O	Giriş	1bit
Matris Girişi	AXI4-Stream Video	Giriş	64bit
Matris Çıkışı	AXI4-Stream Video	Çıkış	64bit

Yön girişi arayüzü, 2D FFT dönüşümünün yönünün seçildiği 1bit'lik giriş arayüzüdür. Yön giriş arayüzü düşük lojik seviye olduğu durumda dönüşüm 2D FFT, yüksek lojik seviye olduğu durumda 2D IFFT yönündedir.

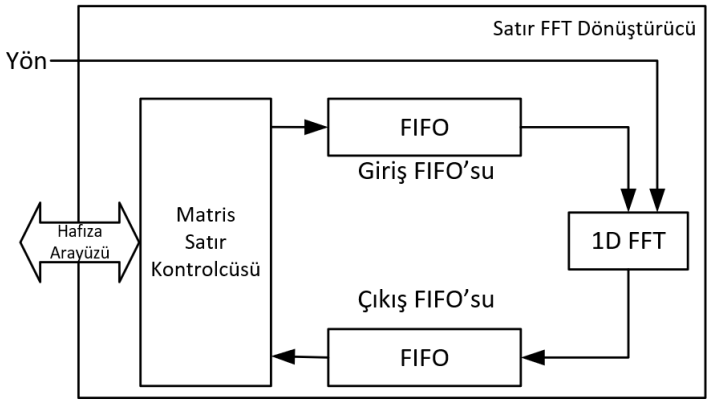
Tablo 4.12. Matris Giriş ve Matris Çıkış arayüzü veri içeriği

Sinyal/Alan Adı	Veri Tipi	Byte	Açıklama
Gerçek sayı	float	4	Karmaşık sayı gerçek sayı bileşeni
Sanal sayı	float	4	Karmaşık sayı sanal sayı bileşeni

Matris giriş ve Matris Çıkış arayüzleri kullanılarak elemanları karmaşık sayı olan matris girişi ve matris çıkışı yapılmaktadır. Matris giriş ve matris çıkış arayüzleri tdata sinyali veri içeriği Tablo 4.12'de yer almaktadır.

Şekil 4.19'daki 2D FFT dönüştürücü donanım mimarisinde, giriş matrisini ve sonuç matrisini depolamak için bir matris tamponu bulunmaktadır. Matris tamponuna elemanları karmaşık sayı olan ve boyutları arama penceresi boyutları ile aynı olan bir matris kaydedilmektedir. Dolayısıyla matris tamponuna kaydedilecek matrisin her elemanı 64bit uzunluğunda ve boyutları 128 × 128'dir. Bu değerler dikkate alındığında matris tamponu boyutu 1.048mb olarak hesaplanmıştır.

2D FFT dönüştürücü donanım mimarisinde bulunan tampon yazıcı donanımı, matris giriş arayüzünden gelen matrisi, matris tamponuna yazmaktadır.2D FFT dönüştürücü donanım mimarisinde bulunan satır FFT dönüştürücü donanımı, 2D FFT dönüşümü işleminin ilk adımı olan satır FFT dönüşümünü yapmaktadır. Satır FFT dönüştürücü donanım mimarisi, Şekil 4.20'de verilmiştir.



Şekil 4.20. Satır FFT Dönüştürücü donanım mimarisi

Satır FFT dönüştürücü donanım mimarisi içerisinde bulunan 1D FFT donanımı için Vivado HLS'in kütüphane olarak desteklediği ve Xilinx firmasının çözüm olarak sunduğu FFT IP'si kullanılmıştır.

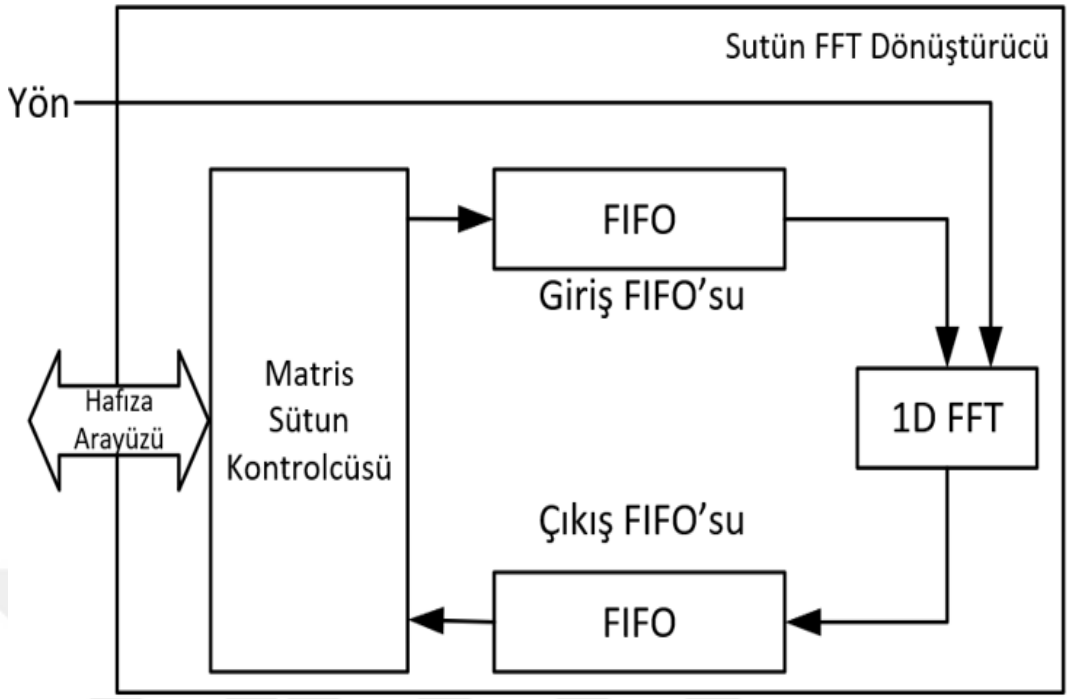
Xilinx FFT IP'si Cooley-Tukey algoritmasını kullanmakta, fixed point ve float veri tipleri kullanacak şekilde yapılandırılabilmekte ve 8 – 65536 noktalı FFT dönüşümü yapabilmektedir. Xilinx FFT IP'si ters ve ileri yönde FFT dönüşümü yapabilmektedir ve dönüşüm yönü 1bit'lik giriş arayüzü ile kontrol edilebilmektedir [27].

Xilinx FFT IP'si FIFO giriş ve FIFO çıkış arayüzlerine sahiptir. FFT'si alınacak diziye giriş FIFO' su üzerinden erişilmekte ve FFT dönüşüm katsayıları çıkış FIFO 'suna yazılmaktadır. Matris tamponunda, elemanları karmaşık sayı olan ve boyutu 128 × 128 arama penceresi boyutlarında olan bir matris tutulmaktadır. Dolayısıyla 2D FFT dönüşümü yapılacak olan matrisin bir satırında 128 nokta bulunmaktadır. Bu nedenle Xilinx FFT IP'si 128 noktalı FFT dönüşümü yapacak şekilde yapılandırılmıştır. Giriş ve çıkış FIFO 'ları ise 128 noktalı veri tutabilecek şekilde yapılandırılmıştır.

Matris tamponuna kaydedilmiş olan giriş matrisinin satırları giriş FIFO' suna yazılmakta, ardından giriş FIFO' su üzerinden erişilen noktalar kullanılarak FFT dönüşümü yapılmaktadır. FFT dönüşümü sırasında hesaplanan dönüşüm katsayıları çıkış FIFO' suna yazılmaktadır.

FFT dönüşümü bitirildikten sonra ve çıkış FIFO' su doldurulduktan sonra dönüşüm katsayıları çıkış FIFO 'su üzerinden okunmaktadır ve sütun FFT dönüşümünde kullanılmak üzere tekrardan matris tamponuna yazılmaktadır. FFT dönüşüm yönü, yön giriş arayüzü ile giriş yapılan 1bit'lik veri doğrultusunda ileri veya ters yönde yapılabilmektedir. Matris tamponunda bulunan 128 satırlı matrisin satır FFT dönüşümü için 128 adet FFT dönüşümü yapılmaktadır.

Satırların FFT dönüşümü yapılmasının ardından, 2D FFT dönüşümünün ikinci adımı olan sütunların FFT dönüşümü yapılmaktadır. Sütunların FFT dönüşümü, Sütun FFT dönüştürücüsü tarafından yapılmaktadır. Şekil 4.21'de Sütun FFT dönüştürücü donanım mimarisi yer almaktadır.



Şekil 4.21. Sütun FFT Dönüştürücü donanım mimarisi

Sütun FFT dönüştürücü donanım mimarisi, temel olarak Satır FFT dönüştürücü donanım mimarisi ile aynı özellikleri göstermektedir. Aralarında bulunan ayırt edici fark, giriş FIFO' sunun matris satırları yerine matris sütunları ile doldurulmasıdır ve çıkış FIFO' sunun matris tamponundaki matrisin sütunları üzerine yazılmasıdır. Sütun FFT dönüştürücü donanımında Satır FFT dönüştürücü donanımında olduğu gibi Xilinx FFT IP'si kullanılmıştır. Matris tamponunda bir sütunu 128 elemana sahip matris bulunması nedeniyle, Sütun FFT dönüştürücü donanımındaki FFT IP'si 128 noktalı FFT dönüşümü yapacak şekilde yapılandırılmıştır. Sütun FFT dönüşümü sırasında matris tamponundan okunan matris sütunu, giriş FIFO'su üzerine yazılmaktadır. Giriş FIFO 'su üzerinden erişilen noktalar kullanılarak FFT dönüşümü yapılmaktadır ve dönüşüm katsayıları çıkış FIFO'suna yazılmaktadır. FFT dönüşümünün bitmesi ile doldurulan çıkış FIFO'su üzerindeki dönüşüm katsayıları, matris tamponu üzerine matris sütunu olarak tekrardan yazılmaktadır. Matris tamponunda bulunan 128 sütunlu matrisin, sütun FFT dönüşümü için 128 adet FFT dönüşümü yapılmaktadır. Sütun FFT dönüşümünün yapılmasının ardından 2D FFT sonuç matrisi üretilmiş ve matris tamponuna kaydedilmiş olmaktadır. 2D FFT dönüşümünün son adımı olarak tampon okuyucu donanımı çalışmaktadır. Tampon okuyucu donanımı matris tamponundan sonuç matrisini okuyup AXI4-Stream Video arayüzü ile çıkış olarak vermektedir.

Şekil 4.19'da donanım mimarisi verilen 2D FFT dönüştürücü donanımı, Vivado HLS ortamında, C++ dili kullanılarak FPGA donanımı olarak tasarlanmıştır. Tasarlanan 2D FFT dönüştürücü FPGA donanımı, Vivado HLS ortamında sentezlenmiştir. Sentez sonuçlarına Tablo 4.13'de yer verilmiştir.

Tablo 4.13. 2D FFT dönüştürücü FPGA donanımı sentez sonuçları

BRAM_18K	DSP48E	FF	LUT	URAM
74	24	14757	22734	0

2D FFT dönüştürücü donanımı zamanlama çıktıları Tablo 4.14'de verilmiştir.

Tablo 4.14. 2D FFT dönüştürücü donanımı zamanlama çıktıları

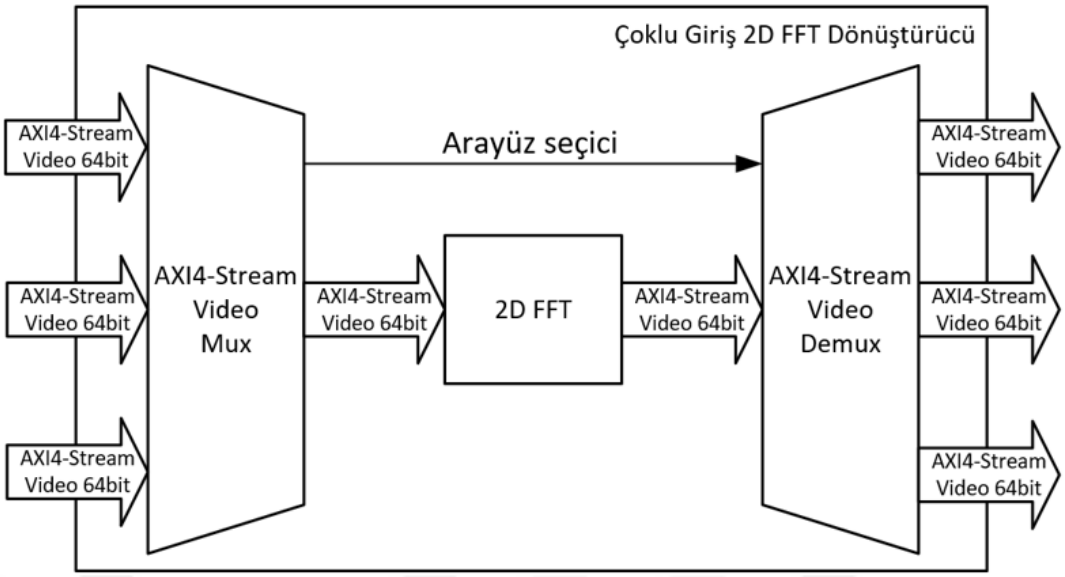
Hedef Saat Frekansı	Kestirilen Saat Frekansı (f_{max})	Belirsizlik	Gerekli Saat Sinyali ($tick_{max}$)
100Mhz	114.285Mhz	1Mhz	190983

2D FFT dönüştürücü donanımı 100Mhz bir saat frekansı girişi ile birlikte çalıştırıldığında, 128 × 128 boyutlarındaki bir matrisin 2D FFT dönüşüm işlemini 1.90ms sürede yapabilmektedir.

4.1.5. Çoklu Giriş-Çoklu Çıkış 2D FFT Dönüştürücü donanımı tasarımı

Şekil 4.2'deki MOSSE video nesne takip algoritması donanım mimarisinde video önişleme, filtre güncelleme ve korelasyon hesaplama adımlarında 2D FFT dönüştürücü donanımı kullanılmıştır. Bu üç işlem adımından video ön işleme ve korelasyon hesaplama adımları tek AXI4-Stream Video arayüzü ile birbirlerine bağlanmıştır ve bu işlem adımları aynı anda çalışmaktadır. Filtre güncelleme, 2D gauss dağılım hesaplama ve video ön işleme adımları ise birbirlerinden ayrık video arayüzlerinde çalışmakta ve bu adımlar aynı anda çalışmamaktadır. Bu doğrultuda, kaynak tüketimini azaltmak için video ön işlem, 2D gauss dağılım hesaplama ve filtre güncelleme adımlarında, tek 2D FFT dönüştürücü donanımı kullanılmıştır.

2D FFT dönüştürücü donanımının, üç işlem adımında beraber kullanılması için 2D FFT dönüştürücü donanımı, Çoklu Giriş-Çoklu Çıkış 2D FFT dönüştürücü (MIMO 2D FFT) donanımı olarak özelleştirilmiştir. Şekil 4.22'de Çoklu Giriş-Çoklu Çıkış 2D FFT dönüştürücü donanım mimarisi verilmiştir.



Şekil 4.22. MIMO 2D FFT dönüştürücü donanım mimarisi

Çoklu Giriş - Çoklu Çıkış 2D FFT dönüştürücü donanımı temel olarak 2D FFT donanımını bulundurmaktadır. 2D FFT dönüştürücü donanımı Çoklu Giriş - Çoklu Çıkış olarak özelleştirmek için AXI4-Stream Video Mux ve AXI4-Stream Video Demux donanımları kullanılmıştır.

AXI4-Stream Video Mux donanımı, üç tane giriş ve bir tane çıkış AXI4-Stream Video arayüzlerine sahiptir. AXI4-Stream Video Mux donanımı, giriş arayüzlerinden SOF sinyali yükselen kenar olan arayüzü çıkış olarak vermektedir. Çıkış olarak verilen arayüzü belirtmek için arayüz seçici çıkışı üzerinden 2 bitlik çıkış verilmektedir. AXI4-Stream Video Mux donanımı giriş çıkış arayüzleri Tablo 4.15'de verilmiştir.

Tablo 4.15. AXI4-Stream Video Mux donanımı giriş çıkış arayüzleri

Arayüz İsmi	Arayüz Türü	Giriş/Çıkış Türü	Veri Boyutu
Giriş 1	AXI4-Stream Video	Giriş	64 bit
Giriş 2	AXI4-Stream Video	Giriş	64bit
Giriş 3	AXI4-Stream Video	Giriş	64bit
Arayüz Seçici	I/O	Çıkış	2bit
Çıkış	AXI4-Stream Video	Çıkış	64bit

AXI4-Stream Video Demux donanımı, bir tane giriş ve üç tane çıkış AXI4-Stream Video arayüzlerine sahiptir. AXI4-Stream Video Demux donanımı, giriş arayüzünden okunan veriyi arayüz seçici girişi doğrultusunda çıkış arayüzünü seçerek, çıkış olarak

vermektedir. AXI4-Stream Video Demux donanımı giriş çıkış arayüzleri Tablo 4.16'de verilmiştir.

Tablo 4.16. AXI4-Stream Video Demux donanımı giriş çıkış arayüzleri

Arayüz İsmi	Arayüz Türü	Giriş/Çıkış Türü	Veri Boyutu
Giriş	AXI4-Stream Video	Giriş	1 bit
Arayüz Seçici	I/O	Giriş	64bit
Çıkış 1	AXI4-Stream Video	Çıkış	64bit
Çıkış 2	AXI4-Stream Video	Çıkış	64bit
Çıkış 3	AXI4-Stream Video	Çıkış	64bit

Şekil 4.22'de donanım mimarisi verilen Çoklu Giriş- Çoklu Çıkış 2D FFT dönüştürücü donanımı, Vivado HLS ortamında C++ dili kullanılarak FPGA donanımı olarak tasarlanmıştır. Tasarlanan Çoklu Giriş-Çoklu Çıkış 2D FFT dönüştürücü FPGA donanımı, Vivado HLS ortamında sentezlenmiştir. Sentez sonuçlarına Tablo 4.17'de yer verilmiştir.

Tablo 4.17. MIMO 2D FFT dönüştürücü FPGA donanımı sentez sonuçları

BRAM_18K	DSP48E	FF	LUT	URAM
74	24	15288	24046	0

Çoklu Giriş- Çoklu Çıkış 2D FFT dönüştürücü donanımı zamanlama çıktıları Tablo 4.18'de verilmiştir. Çoklu Giriş- Çoklu Çıkış 2D FFT dönüştürücü donanımı 100Mhz bir saat frekansı ile çalıştırıldığında 128 × 128 boyutlarındaki bir matrisin 2D FFT dönüşüm işlemini 1.90855ms sürede yapabilmekte ve @523.95fps hızında dönüşüm yapabilmektedir.

Tablo 4.18. MIMO 2D FFT dönüştürücü donanımı zamanlama çıktıları

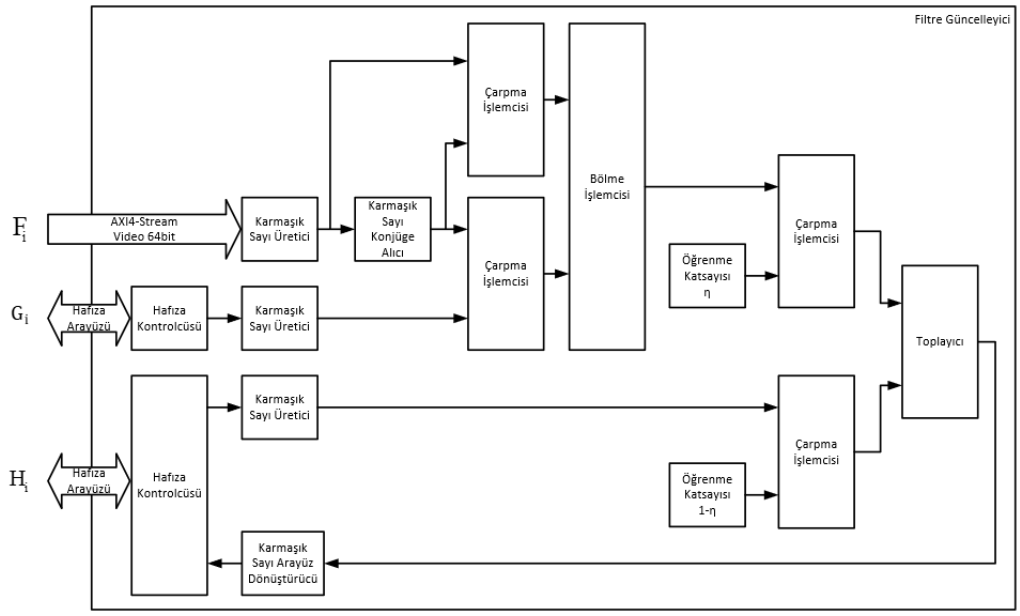
Hedef Saat Frekansı	Kestirilen Saat Frekansı (f_{max})	Belirsizlik	Gerekli Saat Sinyali ($tick_{max}$)
100Mhz	114.285Mhz	1Mhz	190855

4.1.6. Filtre Güncelleyici donanım tasarımı

Filter güncelleyici donanımı Şekil 4.2'deki MOSSE video nesne takip algoritması donanım mimarisinde filtre güncelleme adımında, korelasyon filtresinin güncellenmesi amacıyla kullanılmıştır. [10]'da MOSSE video nesne takip algoritması

korelasyon filtresi güncellenmesi adımında denklem (2.7) kullanılarak A_i ve B_i filtre katsayı matrisleri türetilmiştir. Korelasyon filtresinin güncellenmesi için filtre katsayı matrisleri denklem (2.8) ile beraber kullanılmıştır.

Korelasyon filtresinin güncellenmesi adımında, A_i ve B_i filtre katsayı matrislerinin denklem (2.8)'de kullanılmadan önce denklem (2.9) ve denklem (2.10) kullanılarak güncellenmesi gerekmektedir. Filtre katsayı matrislerinin, denklem (2.9) ve denklem (2.10)'daki gibi öğrenme katsayısı kullanılarak güncellenebilmesi için iki ayrı matris tamponu kullanılarak matrislerin eski değerlerinin kaydedilmesi gereksinimini ortaya çıkarmaktadır. İki ayrı matris bileşeninin matris tamponu ile kaydedilmesi fazladan kaynak tüketimine sebebiyet vermektedir. Bu durum dikkate alındığında, korelasyon filtresinin güncellenmesi işleminde, türetilmemiş denklem (2.7) kullanılmıştır. Korelasyon filtresinin güncellenmesi adımında denklem (2.7)'nin kullanımı H_i korelasyon filtresi matrisini matris tamponunda kaydetme gereksinimi oluşturmakta ve A_i , B_i filtre katsayı matrislerinin matris tamponunda tutulması gereksinimine göre daha az kaynak tüketimi sağlamaktadır. Filtre güncelleyici donanım mimarisi Şekil 4.23'de verilmiştir.



Şekil 4.23. Filtre Güncelleyici donanım mimarisi

Filtre güncelleyici donanımına, F_i Matrisi Girişi arayüzü ile ön işlenmiş ve frekans boyutuna dönüştürülmüş arama penceresi matrisi giriş yapılmıştır. Bir önceki adımda

hesaplanan korelasyon filtresinin öğrenme katsayısı ile beraber filtre güncellenmesinde kullanılması için filtre tamponunda kayıtlı olan filtre matrisine, filtre matris tamponu arayüzü ile erişilmiştir. Korelasyon filtresi güncellendikten sonra filtre matris tamponu arayüzü ile matris tamponuna yazılmıştır. Filtre güncelleme adımında kullanılan 2 boyutlu gauss dağılım matrisinin frekans boyutuna dönüştürülmüş hali olan G_i matrisine, matris tamponu üzerinden hafıza arayüzü ile erişilmiştir. Filtre güncelleyici donanımı giriş çıkış arayüzlerine Tablo 4.19'de yer verilmiştir.

Tablo 4.19. Filtre güncelleyici donanımı giriş çıkış arayüzleri

Arayüz İsmi	Arayüz Türü	Giriş/Çıkış Türü	Veri Boyutu
F_i Matris Arayüzü	AXI4-Stream Video	Giriş	64bit
G_i Matris Arayüzü	BRAM Arayüzü	Giriş	64bit
H_i Matris Arayüzü	BRAM Arayüzü	Giriş/Çıkış	64bit

Tablo 4.20. F_i Matris Arayüzü veri içeriği

Sinyal/Alan Adı	Veri Tipi	Byte	Açıklama
Gerçek sayı	float	4	Karmaşık sayı gerçek sayı bileşeni
Sanal sayı	float	4	Karmaşık sayı sanal sayı bileşeni

Tablo 4.21. G_i Matris Arayüzü veri içeriği

Sinyal/Alan Adı	Veri Tipi	Byte	Açıklama
Gerçek sayı	float	4	Karmaşık sayı gerçek sayı bileşeni
Sanal sayı	float	4	Karmaşık sayı sanal sayı bileşeni

Tablo 4.22. H_i Matris Arayüzü veri içeriği

Sinyal/Alan Adı	Veri Tipi	Byte	Açıklama
Gerçek sayı	float	4	Karmaşık sayı gerçek sayı bileşeni
Sanal sayı	float	4	Karmaşık sayı sanal sayı bileşeni

Filtre güncelleyici donanımı arayüzleri ile erişilen ve filtre güncelleme işleminde kullanılacak olan veriler karmaşık sayı kümesi içerisindedir. Karmaşık sayı ifadelerin bir matris tamponunda tutulması veya AXI4-Stream arayüzü ile iletilmesi için gerçek ve sanal kısımlar birleştirilerek 64bit uzunluğunda bir değişken türüne dönüştürülmüştür. Bu doğrultuda, filtre güncelleyici donanım arayüzleri ile erişilen veriler ilk olarak Vivado HLS ortamının desteklediği, karmaşık sayı değişken türüne dönüştürülmüştür. Vivado HLS ortamı, desteklediği karmaşık sayı değişken türü ile toplama, çıkarma, çarpma ve bölme işlemlerini desteklemektedir. Donanım arayüzleri

ile erişilen veriler, filtre güncelleyici donanım mimarisinde bulunan, karmaşık sayı üretici modülü ile Vivado HLS ortamının desteklediği karmaşık sayı veri tipine dönüştürülmüştür. Filtre güncelleme işleminin ardından üretilen filtre matrisi elemanları, karmaşık sayı arayüz dönüştürücü modülü ile arayüzlerden iletilecek halde olan 64bit uzunluğundaki veri tipine dönüştürülmüştür. Karmaşık sayı elemanların gerçek sayı ve sanal sayı kısımları, IEEE754 standartı ile float veri tipinden integer veri tipine dönüştürülmüştür. Filtre güncelleyici donanımı arayüzleri veri içerikleri Tablo 4.20, Tablo 4.21 ve Tablo 4.22'de verilmiştir.

Denklem (2.7) bulunan F_i^* matrisi, F_i matrisinin konjügesi alınmış halidir ve filtre güncelleyici donanım mimarisinde karmaşık sayı konjüge alıcı modülünde hesaplanmaktadır. Karmaşık sayı konjüge alıcı modülü, Vivado HLS ortamında kullanılan karmaşık sayı veri tipinin sanal kısmını -1 ile çarpmaktadır.

Filtre güncelleyici donanım mimarisinde bulunan çarpma işlemcisi, bölme işlemcisi ve toplayıcı modülleri, Vivado HLS ortamının desteklediği karmaşık sayı işlem modülleridir.

Denklem (2.7) kullanılan öğrenme katsayıları, filtre güncelleyici donanım mimarisinde 32bit kaydedicilerde tutulmuştur. Filtre güncelleyici donanım mimarisinde öğrenme katsayısı 0.125 olarak seçilmiştir.

Şekil 4.23'de donanım mimarisi verilen filtre güncelleyici donanımı, Vivado HLS ortamında C++ kullanılarak FPGA donanımı olarak tasarlanmıştır. Tasarlanan filtre güncelleyici FPGA donanımı Vivado HLS ortamında sentezlenmiştir. Filtre güncelleyici donanımı FPGA donanımı sentez sonuçlarına Tablo 4.23'da yer almaktadır.

Tablo 4.23. Filtre güncelleyici FPGA donanımı sentez sonuçları

BRAM_18K	DSP48E	FF	LUT	URAM
0	5	2240	2308	0

Vivado HLS ortamında sentezlenen ön işlemci donanımı zamanlama çıktıları Tablo 4.24'de yer almaktadır.

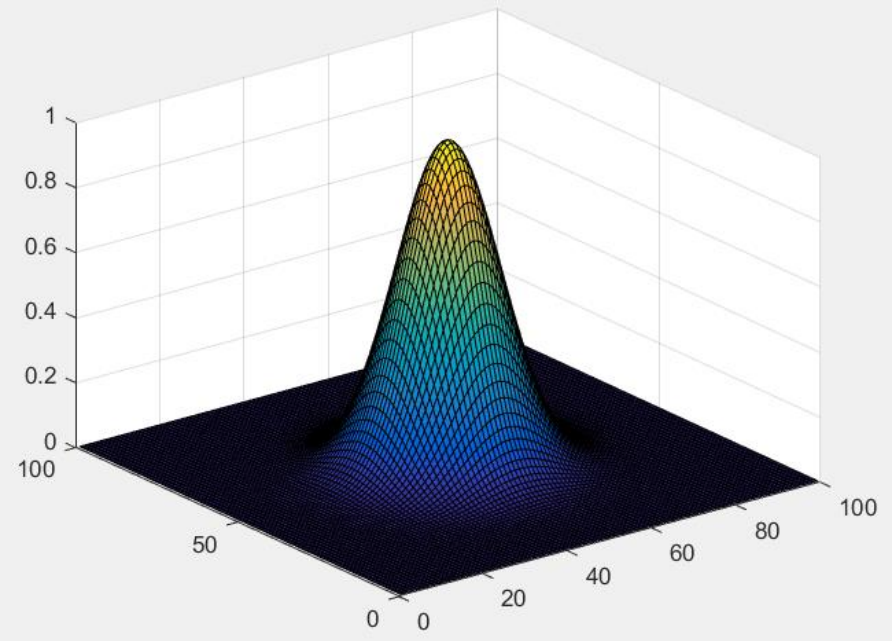
Tablo 4.24. Ön İşlemci donanımı zamanlama çıktıları

Hedef Saat Frekansı	Kestirilen Saat Frekansı (f_{max})	Belirsizlik	Gerekli Saat Sinyali ($tick_{max}$)
100Mhz	114.233Mhz	1Mhz	262174

Filtre güncelleyici donanımı 100Mhz saat frekansı ile beslendiğinde, 128 × 128 boyutlarındaki korelasyon filtresi matrisini @381.42fps ile güncelleyebilmektedir.

4.1.7. 2D Gauss Dağılım Hesaplayıcı donanımı tasarımı

2D Gauss dağılım hesaplayıcı donanımı, korelasyon filtresinin ilklendirilmesi aşamasında kullanılmaktadır. Korelasyon filtresinin ilklendirilmesi aşamasında, filtre güncelleme adımında kullanılan, iki boyutlu gauss dağılım matrisi hesaplanmaktadır.

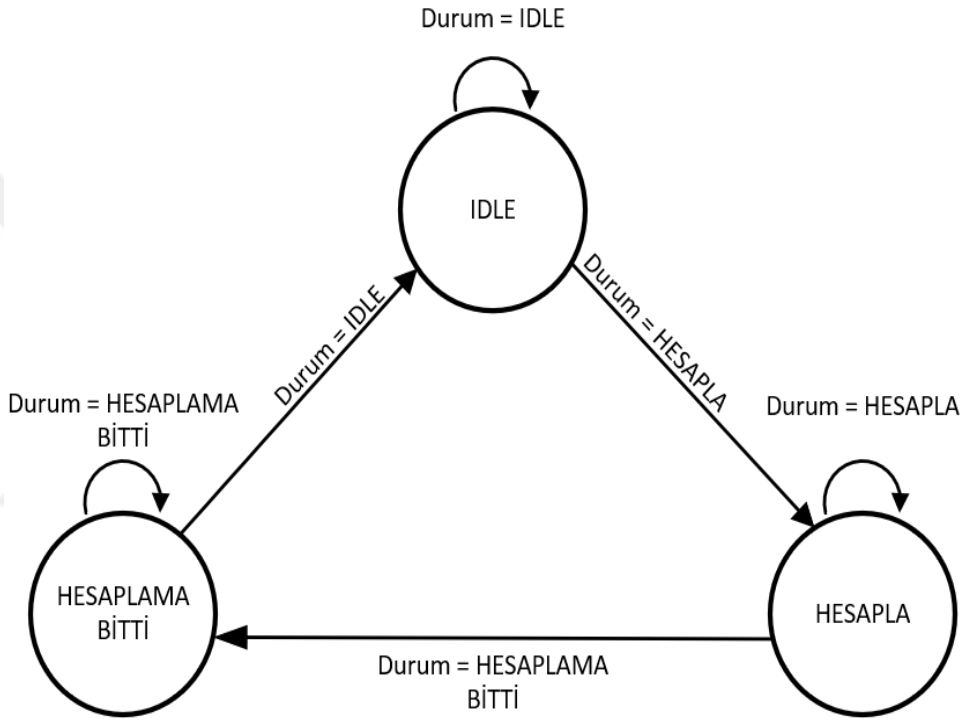


Şekil 4.24. 2D Gauss'yen dağılımı grafiği

İki boyutlu gauss dağılımın hesaplanması için denklem (4.5) kullanılmaktadır. Denklem (4.5)'deki i ve j, gauss dağılım matrisinin eleman satır ve sütun numaralarını, w ve h ise arama penceresi uzunluk ve genişlik değerlerini ifade etmektedir. Denklem (4.5)'deki sigma değeri için sabit 100 değeri kullanılmaktadır.

$$e^{-\left(\frac{\left(i-\frac{w}{2}\right)^2+\left(j+\frac{h}{2}\right)^2}{2\sigma}\right)} \tag{4.5}$$

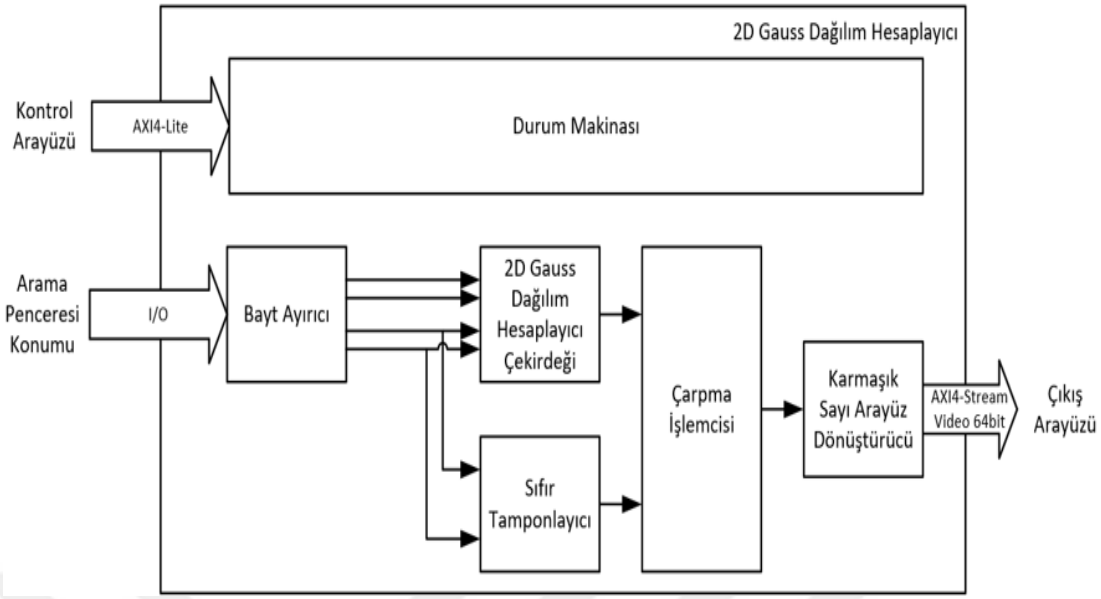
2D gauss dağılım hesaplayıcı donanımı durum makinası Şekil 4.25'de yer almaktadır. Başlangıç aşamasında IDLE durumunda olan 2D gauss dağılım hesaplayıcı donanımı, kontrol arayüzü ile HESAPLA durumuna geçirilmektedir. HESAPLA durumunda, denklem (4.5) kullanılarak 2D gauss dağılım matrisi hesaplanmaktadır ve çıkış olarak verilmektedir. 2D gauss dağılım matrisinin hesaplanmasının ardından HESAPLAMA BİTTİ durumuna geçiş yapılmaktadır. HESAPLAMA BİTTİ durumundan kontrol arayüzü ile IDLE durumuna geçiş yapılmaktadır.



Şekil 4.25. 2D Gauss Dağılım Hesaplayıcı Donanımı Durum Makinası

2D gauss dağılım hesaplayıcı donanım mimarisine, Şekil 4.26'de yer verilmiştir. Tablo 4.25'de 2D gauss dağılım hesaplayıcı donanımı giriş çıkış arayüzlerine yer verilmiştir. AXI4-Lite Slave kontrol arayüzü ile durum geçiş kontrolü yapılabilmektedir ve bulunulan durum bilgisi okunabilmektedir. Arama penceresi konumu arayüzü giriş yapılan 4 bayt uzunluğundaki giriş verisi içerisinde arama penceresi konumu ve boyutları bilgilerini içermektedir.

Arama penceresi konumu ve boyutu bilgisi, sıfır tamponlayıcı ve 2D gauss dağılım hesaplayıcı çekirdeği tarafından kullanılmak üzere arama penceresi konumu arayüzü verisinden ayrıştırılmıştır.



Şekil 4.26. 2D Gauss Dağılım Hesaplayıcı donanım mimarisi

Tablo 4.25. 2D gauss dağılım hesaplayıcı donanımı giriş çıkış arayüzleri

Arayüz İsmi	Arayüz Türü	Giriş/Çıkış Türü	Veri Boyutu
Kontrol Arayüzü	AXI4-Lite Slave	Giriş/Çıkış	8bit
Arama Penceresi Konumu	I/O	Giriş	64bit
Çıkış Arayüzü	AXI4-Stream Video	Çıkış	64bit

2D gauss dağılım hesaplayıcı donanımı, arama penceresi boyutları ile aynı boyutta olan 128 × 128 boyutlarında çıkış matrisi hesaplamaktadır. Arama penceresinin boyutunun 128 × 128 boyutundan küçük olduğu durumlarda çıkış matrisinin arta kalan kısımları sıfır tamponlanmıştır. 2D gauss dağılım hesaplayıcı çekirdeği modülü denklem (4.5) işlemini hesaplamaktadır. Denklem (4.5)'de bulunan eksponansiyel işlemi için Vivado HLS tarafından desteklenen matematik kütüphanesi kullanılmıştır. Hesaplanan ve sıfır tamponlanan 2D gauss dağılım matrisi, karmaşık sayı arayüz dönüştürücü modülü ile arayüz veri tipine dönüştürülmüştür ve AXI4-Stream Video matris çıkış arayüzü ile çıkış olarak verilmiştir.

Şekil 4.26'de donanım mimarisi verilen 2D gauss dağılım hesaplayıcı donanımı Vivado HLS ortamında C++ dili kullanılarak FPGA donanımı olarak tasarlanmıştır. Tasarlanan 2D gauss dağılım hesaplayıcı FPGA donanımı, Vivado HLS ortamı kullanılarak sentezlenmiştir. 2D gauss dağılım hesaplayıcı donanımı, FPGA donanımı sentez sonuçlarına Tablo 4.26'da yer verilmiştir.

Tablo 4.26. Filtre güncelleyici FPGA donanımı sentez sonuçları

BRAM_18K	DSP48E	FF	LUT	URAM
0	26	3343	6500	0

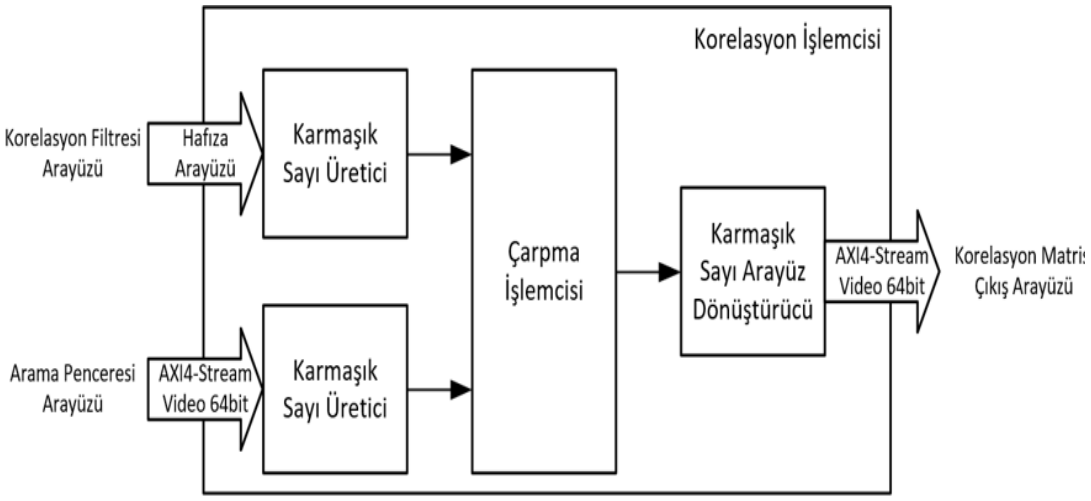
2D gauss dağılım hesaplayıcı donanımı zamanlama çıktıları Tablo 4.27'de yer almaktadır.

Tablo 4.27. Ön İşlemci donanımı zamanlama çıktıları

Hedef Saat Frekansı	Kestirilen Saat Frekansı (f_{max})	Belirsizlik	Gerekli Saat Sinyali ($tick_{max}$)
100Mhz	114.233Mhz	1Mhz	262174

4.1.8. Korelasyon Hesaplama İşlemcisi donanım tasarımı

Korelasyon hesaplama işlemcisi donanımı, Şekil 4.2'deki MOSSE video nesne takip algoritması donanım mimarisinde, korelasyon hesaplama adımında kullanılmıştır. Korelasyon hesaplama işlemcisi, filtre tamponundan erişilen korelasyon filtresini ve arama penceresinin ön işlenmiş ve frekans uzayına dönüştürülmüş halini kullanarak korelasyon matrisini hesaplamaktadır. Korelasyon hesaplama işlemcisi donanım mimarisi Şekil 4.27'de yer almaktadır.



Şekil 4.27. Korelasyon hesaplama işlemcisi donanım mimarisi

Korelasyon hesaplama işlemcisi, korelasyon filtresi matrisini ve frekans uzayındaki arama penceresi matrisini eleman bazlı çarparak korelasyon matrisini hesaplamaktadır. Korelasyon hesaplama işlemcisi giriş çıkış arayüzlerine, Tablo 4.28'de yer verilmiştir. Korelasyon hesaplama işlemcisi korelasyon filtresine filtre

tamponu üzerinden hafıza arayüzü ile erişmektedir. Frekans uzayındaki arama penceresi matrisine, AXI4-Stream Video arayüzü kullanılarak erişilmektedir.

Tablo 4.28. Korelasyon hesaplama işlemcisi donanımı giriş çıkış arayüzleri

Arayüz İsmi	Arayüz Türü	Giriş/Çıkış Türü	Veri Boyutu
Korelasyon Filtresi Arayüzü	BRAM Arayüzü	Giriş	64bit
Arama Penceresi Arayüzü	AXI4-Stream Video	Giriş	64bit
Korelasyon Matrisi Çıkış Arayüzü	AXI4-Stream Video	Çıkış	64bit

Giriş arayüzleri ile erişilen 64bit uzunluğundaki verilerin, Vivado HLS tarafından desteklenen karmaşık sayı değişken türüne dönüştürülmesi için karmaşık sayı üreteci donanımı kullanılmıştır. Karmaşık sayı üretici modülleri, Tablo 4.29'da yer verilen giriş çıkış arayüzleri veri içerikleri doğrultusunda, karmaşık sayı dönüşümünü gerçekleştirmektedir.

Tablo 4.29. Korelasyon hesaplama işlemcisi giriş çıkış arayüzleri veri içeriği

Sinyal/Alan Adı	Veri Tipi	Byte	Açıklama
Gerçek sayı	float	4	Karmaşık sayı gerçek sayı bileşeni
Sanal sayı	float	4	Karmaşık sayı sanal sayı bileşeni

Korelasyon hesaplama işlemcisi donanım mimarisinde bulunan çarpma işlemcisi, frekans uzayındaki korelasyon filtresi ve frekans uzayındaki arama penceresi matrislerini eleman bazlı çarpmaktadır. Çarpma işlemcisi, Vivado HLS tarafından desteklenen karmaşık sayı veri tiplerini kullanarak karmaşık sayı çarpma işlemi yapmaktadır. Çarpma işlemcisinin girişleri yapılan matrisleri eleman bazlı olarak çarpması ile korelasyon matrisi hesaplanmaktadır. Hesaplanan korelasyon matrisinin elemanları karmaşık sayıdır. Bu doğrultuda, korelasyon matrisi elemanları karmaşık sayı arayüz dönüştürücü modülü ile arayüz veri tipine dönüştürülmektedir ve AXI4-Stream Video korelasyon matrisi çıkış arayüzü ile çıkış olarak verilmektedir.

Şekil 4.27'de donanım mimarisi verilen korelasyon hesaplama işlemcisi donanımı Vivado HLS ortamında C++ dili kullanılarak FPGA donanımı olarak tasarlanmıştır. Tasarlanan korelasyon hesaplama işlemcisi FPGA donanımı Vivado HLS ortamı kullanılarak sentezlenmiştir. Korelasyon hesaplama işlemcisi FPGA donanımı sentez sonuçlarına, Tablo 4.30'da yer verilmiştir.

Tablo 4.30. Korelasyon hesaplama işlemcisi FPGA donanımı sentez sonuçları

BRAM_18K	DSP48E	FF	LUT	URAM
0	16	1938	3181	0

Korelasyon hesaplama işlemcisi donanımı zamanlama çıktıları, Tablo 4.31'de yer almaktadır.

Tablo 4.31. Korelasyon hesaplama işlemcisi donanımı zamanlama çıktıları

Hedef Saat Frekansı	Kestirilen Saat Frekansı (f_{max})	Belirsizlik	Gerekli Saat Sinyali ($tick_{max}$)
100Mhz	111.656Mhz	1Mhz	37121

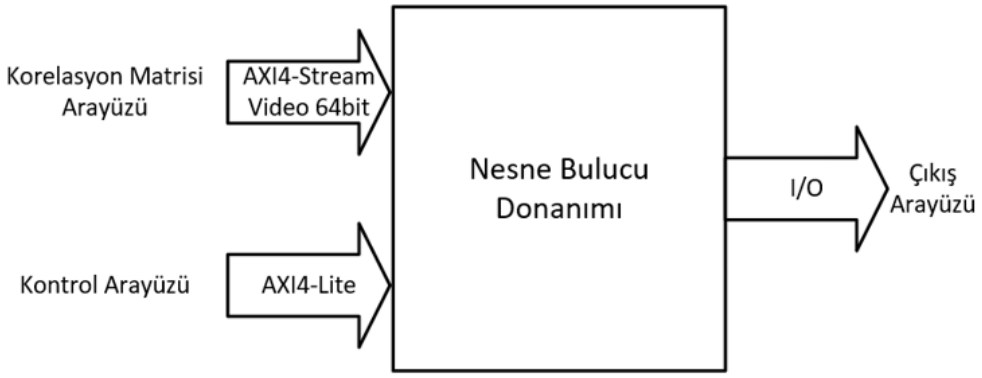
Korelasyon hesaplama işlemcisi donanımı 100Mhz saat frekansı girişi yapıldığında, 128 × 128 boyutlarına sahip korelasyon matrisini @2693.89fps ile hesaplayabilmektedir.

4.1.9. Nesne Konumu Bulucu donanım tasarımı

Nesne konumu bulucu donanımı, Şekil 4.2'deki MOSSE video nesne takip algoritması donanım mimarisinde, korelasyon hesaplama adımında kullanılmıştır. Nesne konumu bulucu donanımı, korelasyon hesaplama işlemcisi donanımı kullanılarak hesaplanan korelasyon matrisinin üzerindeki en büyük korelasyon değerini bulan FPGA donanımıdır.

Korelasyon matrisinin en büyük değerlikli elemanı, korelasyonun en büyük olduğu noktayı temsil etmektedir. Korelasyonun en büyük olduğu noktanın nesne konumunun olduğu nokta olduğu ikinci bölüm başlığı altında açıklanmıştır. Bu doğrultuda nesne konumu bulucu donanımı, giriş arayüzü ile gönderilen korelasyon matrisinin tüm elemanlarını inceleyerek en büyük değerlikli elemanı bulmaktadır ve en büyük değerlikli elemanın korelasyon matrisindeki satır ve sütun numaralarını çıkış olarak vermektedir.

Nesne konumu bulucu donanımı, korelasyon matrisi içerisinde bulunan karmaşık sayı elemanları büyüklüklerini karşılaştırmaktadır. Karşılaştırma işlemi sırasında giriş matrisinin karmaşık sayı elemanlarının modülleri hesaplanmaktadır ve hesaplanan modüller kullanılarak en büyük değerlikli matris elemanı bulunmaktadır.



Şekil 4.28. Nesne Konumu Bulucu Donanımı

Nesne takip edici donanımının ilklendirilmesi aşamasında, 2D gauss dağılım üretici, ön işlemci ve nesne çerçeveleme işlemci donanımlarına nesne konumu ve arama penceresi boyutu ilk değerleri giriş olarak verilmesi gerekmektedir. Nesne konumu ve arama penceresinin ilklendirilmesi için nesne konumu bulucu donanımı kullanılmaktadır. Nesne konumu ve arama pencere ilk değerleri nesne bulucu donanımı AXI4-Lite slave kontrol arayüzü ile ilklendirilmektedir. Nesne konumu bulucu donanımı giriş çıkış arayüzlerine, Tablo 4.32'de yer verilmiştir.

Tablo 4.32. Nesne Konumu Bulucu donanımı giriş çıkış arayüzleri

Arayüz İsmi	Arayüz Türü	Giriş/Çıkış Türü	Veri Boyutu
Korelasyon Matrisi Arayüzü	AXI4-Stream Video	Giriş	64bit
Kontrol Arayüzü	AXI4-Lite Slave	Giriş	64bit
Çıkış Arayüzü	I/O	Çıkış	32bit

Nesne konumu bulucu donanımına korelasyon matrisi girişi için AXI4-Stream Video korelasyon matrisi giriş arayüzü kullanılmaktadır. Korelasyon matrisinin AXI4-Stream Video arayüzü ile aktarılması ile tüm matris elemanları karşılaştırılmaktadır. Korelasyon matrisinin son elemanında aktarılması ile birlikte, korelasyonun en büyük olduğu nokta bulunmaktadır. Korelasyonun en büyük olduğu noktasının satır sütün bilgisi, I/O arayüzü ile çıkış olarak verilmektedir.

Nesne konumu bulucu donanımı Vivado HLS ortamında C++ dili kullanılarak FPGA donanımı olarak tasarlanmıştır. Tasarlanan Nesne Konumu Bulucu FPGA donanımı Vivado HLS ortamı kullanılarak sentezlenmiştir. Nesne konumu bulucu FPGA donanımı sentez sonuçlarına Tablo 4.33'da yer verilmiştir.

Tablo 4.33. Nesne konumu bulucu FPGA donanımı sentez sonuçları

BRAM_18K	DSP48E	FF	LUT	URAM
0	0	33	151	0

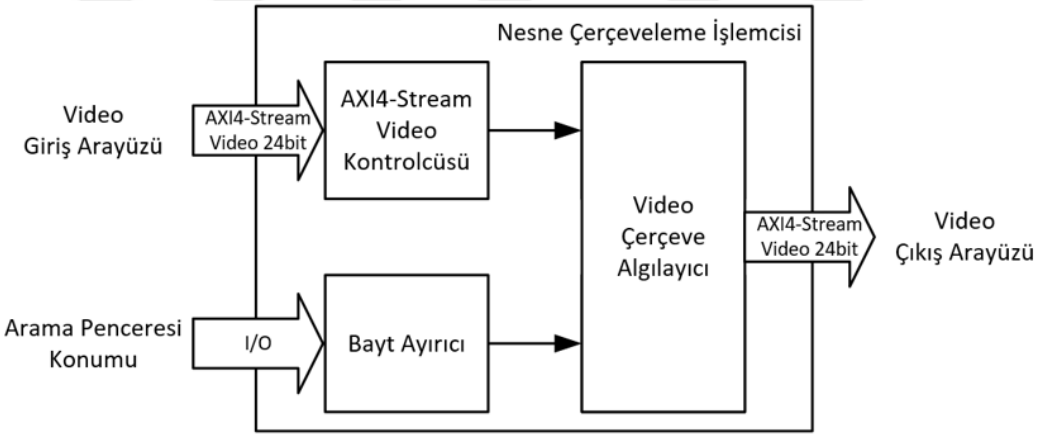
Nesne konumu bulucu donanımı zamanlama çıktıları, Tablo 4.34'de yer almaktadır. Nesne konumu bulucu donanımı 100Mhz saat frekansı ile beslendiğinde, 128 × 128 boyutlarındaki korelasyon matrisinin en büyük elemanının konumunu @6009.254fps ile bulabilmektedir.

Tablo 4.34. Nesne konumu bulucu donanımı zamanlama çıktıları

Hedef Saat Frekansı	Kestirilen Saat Frekansı (f_{max})	Belirsizlik	Gerekli Saat Sinyali ($tick_{max}$)
100Mhz	522.193Mhz	1Mhz	16641

4.1.10. Nesne Çerçeveleme İşlemcisi donanım tasarımı

Nesne çerçeveleme işlemcisi Şekil 4.2'deki MOSSE algoritması FPGA donanım mimarisinde nesne çerçeveleme adımında kullanılmaktadır.



Şekil 4.29. Nesne Çerçeveleme İşlemcisi Donanım Mimarisi

Nesne çerçeveleme işlemcisi, korelasyon hesaplama adımında piksel koordinat düzleminde konumu bulunan nesnenin, video çerçevesi üzerinde nesne konumunu merkez alınacak şekilde çerçevelenmesi işlemini yapmaktadır. Nesne çerçeveleme işlemcisine, video giriş arayüzü kullanılarak işlem yapılacak video girişi yapılmaktadır. Video ön işleme adımında VDMA kullanılarak DDR hafızaya yazılan video çerçevesi, nesne çerçeveleme adımında VDMA kullanılarak DDR hafızadan okunmakta ve nesne çerçeveleme işlemcisine giriş olarak verilmektedir.

Nesne çerçeveleme işlemcisi donanım mimarisi blok diyagramına, Şekil 4.29'da yer verilmiştir. Nesne çerçeveleme işlemcisinde, video giriş arayüzü ile giriş yapılan videonun çerçevelenecek olan piksellerin algılanması için video çerçeve algılayıcı modülü kullanılmaktadır. Video çerçeve algılayıcı modülü, giriş yapılan videonun çerçevelenecek olan piksellerini, arama penceresi konumu arayüzü ile girişi yapılan nesne konumu ve arama penceresi boyutları verileri kullanılarak algılamaktadır. Çerçevelenecek olan pikselin algılanmasının ardından, video çerçeve piksel değeri yerine nesne çerçeve piksel değeri arayüz hattına yazılmaktadır. Bu doğrultuda Şekil 2.6'da görüleceği üzere çerçeveleme işlemi yapılmaktadır. Çerçeveleme işlemi yapılan video çerçevesi, video çıkış arayüzü ile çıkış olarak verilmektedir. Nesne çerçeveleme işlemcisi giriş çıkış arayüzlerine Tablo 4.35'de yer verilmiştir.

Tablo 4.35. Nesne çerçeveleme işlemcisi donanımı giriş çıkış arayüzleri

Arayüz İsmi	Arayüz Türü	Giriş/Çıkış Türü	Veri Boyutu
Video Giriş	AXI4-Stream Video	Giriş	24bit
Arama Penceresi Konumu	I/O	Giriş	32bit
Video Çıkış	AXI4-Stream Video	Çıkış	24bit

Nesne çerçeveleme işlemcisi donanımı Vivado HLS ortamında C++ dili kullanılarak FPGA donanımı olarak tasarlanmıştır. Tasarlanan nesne çerçeveleme işlemcisi FPGA donanımı Vivado HLS ortamı kullanılarak sentezlenmiştir. Nesne çerçeveleme işlemcisi FPGA donanımı sentez sonuçlarına, Tablo 4.36'da yer verilmiştir.

Tablo 4.36. Nesne çerçeveleme işlemcisi FPGA donanımı sentez sonuçları

BRAM_18K	DSP48E	FF	LUT	URAM
0	0	427	1209	0

Nesne çerçeveleme işlemcisi donanımı 100Mhz saat frekansı girişi yapıldığında, yineleme hızı en fazla @325.516fps olacak şekilde 640 × 480 boyutlarındaki RGB video üzerinde çerçeveleme işlemi yapabilmektedir. Nesne çerçeveleme işlemcisi donanımı zamanlama çıktılarına, Tablo 4.37'de yer verilmiştir.

Tablo 4.37. Nesne çerçeveleme donanımı zamanlama çıktıları

Hedef Saat Frekansı	Kestirilen Saat Frekansı (f_{max})	Belirsizlik	Gerekli Saat Sinyali ($tick_{max}$)
100Mhz	136.967Mhz	1Mhz	307204

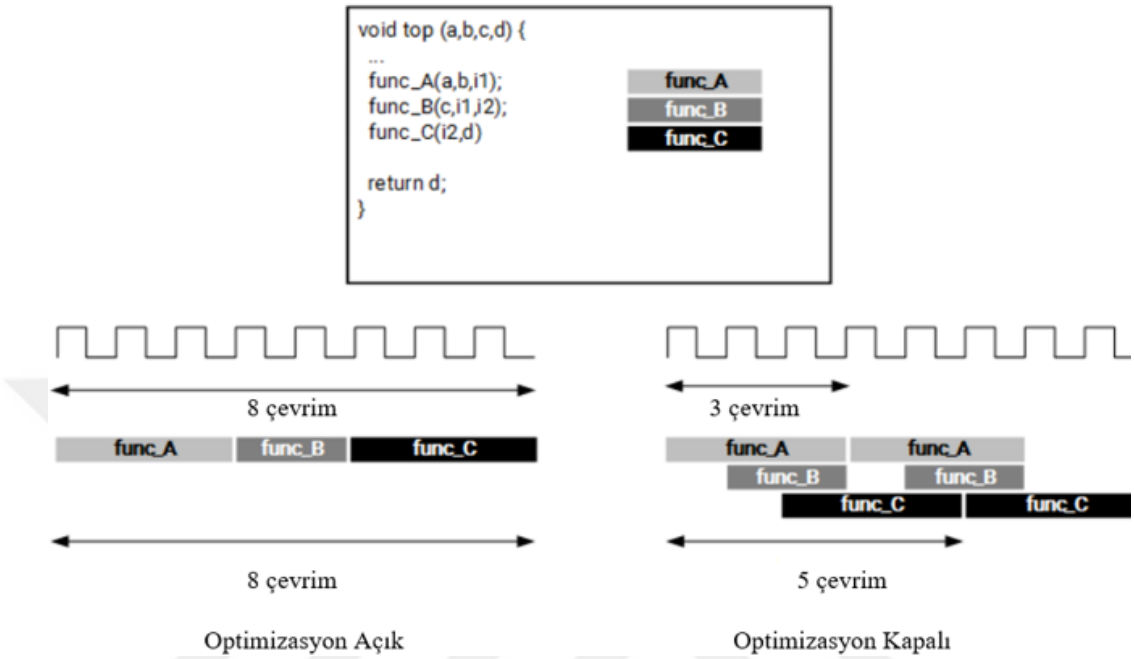
4.2. Donanım Optimizasyon Yöntemleri

Mosse video nesne takip algoritması donanım mimarisinde bulunan donanımların tasarımında, kaynak tüketiminin azaltılması, zaman gecikmelerinin azaltılması ve saat frekansının yükseltilmesi için bazı tasarım optimizasyonları kullanılmıştır. Kullanılan optimizasyon teknikleri, Vivado HLS tarafından desteklenmektedir. Vivado HLS ortamının desteklediği optimizasyonları kullanmak için C dili ile bazı komut satırları yazılmıştır. Kullanılan optimizasyon komutları, donanımdan soyutlanmış bir şekilde optimize edilmiş tasarım yapabilme imkanı sunmaktadır.

4.2.1. Hls dataflow optimizasyonu

Vivado HLS ortamında C++ dili kullanılarak donanım tasarımı yapılmaktadır. C++ dili sıralı işleyen bir dildir. Vivado HLS ortamında optimizasyon komutları kullanılmadan yapılan tasarımlarda veri bağımlılığı nedeniyle gecikmeler oluşmaktadır. Örnek olarak Vivado HLS ortamında C++ dili ile tanımlanan döngüler veya fonksiyonlar, sıralı çalışacak şekilde sentezlenmektedir. Bu durum, ardışık döngü veya fonksiyonların çalışabilmesi için bir önceki döngü veya fonksiyonun bitmesini beklemesine sebep olmakta ve gecikmeleri arttırmaktadır. Dataflow optimizasyonu tanımlandığında, YSS aracı sıralı döngüler veya fonksiyonlar arasındaki veri akışını analiz etmektedir. Aralarında veri akışı tespit edilen döngüler ve fonksiyonlar için veri akışının bekleme olmadan yapılması için veri kanalları oluşturulmaktadır. Bu veri kanalları, ping-pong RAM ve FIFO'lardan oluşturulabilmektedir. Oluşturulan veri kanalları vasıtasıyla sıralı tanımlanan döngüler veya fonksiyonlar arasındaki veri akışı, ilk döngünün veya fonksiyonun çalışmasını bitirmesi beklenmeden yapılabilmektedir. Bu durum,, gecikmeleri azaltmakta ve donanımın daha hızlı çalışmasına katkı sağlamaktadır. Şekil 4.30'da sıralı tanımlanan fonksiyonlar için hls dataflow optimizasyonu kullanımı gösterilmiştir. Şekilde a, b ve c fonksiyonları sıralı çalışmaktadır ve hls dataflow optimizasyonu kullanılmadığında birbirlerinin bitmesini beklediklerinden 8 çevrim süresi beklenmektedir. Hls dataflow optimizasyonu kullanılan tasarımda ise fonksiyonlar arasındaki veri akışı bekleme yapılmaksızın sağlanmaktadır ve gecikmeler azaltılarak fonksiyonlar 3 çevrimde çalıştırılabilmektedir [28].

Hls dataflow optimizasyonu ön işlemci, 2D FFT dönüştürücü, filtre güncelleyici, 2D gauss dağılım hesaplayıcı, korelasyon işlemcisi, nesne konum bulucu ve nesne çerçeveleme işlemcisi donanımlarında kullanılmıştır.



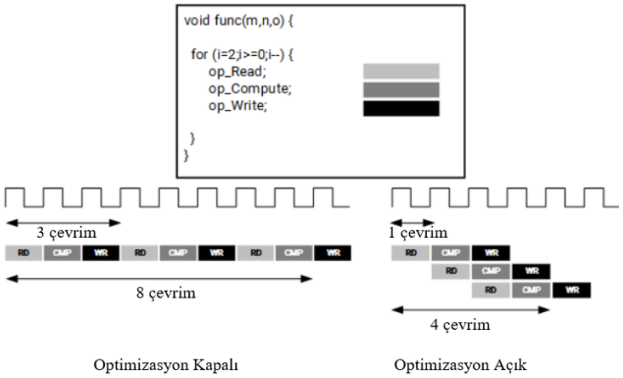
Şekil 4.30. Hls dataflow optimizasyonu kullanımı çevrim süreleri [28]

4.2.2. Hls pipeline optimizasyonu

Vivado HLS ortamında tanımlanan bir döngünün içerisindeki işlemlerin, eş zamanlı bir şekilde çalışmasını sağlayacak şekilde, donanımın optimize edilmesi yöntemidir. Tanımlanan bir döngünün işlem yapmaya başlaması ve işlemi bitirip yeni bir girişi işleme alması arasında geçen çevrim süresi, II ile ifade edilmektedir. Şekil 4.31'de hls pipeline optimizasyonu kullanımı çevrim süreleri gösterilmiştir. Şekilde optimizasyonun kapalı olduğu durumda, yeni bir girişin işlenmesi için 3 çevrim beklenmektedir ve bütün döngünün bitirilmesi için 8 çevrim beklenmektedir. Şekildeki örnekte, döngünün içerisinde verinin okunması, verinin işlenmesi ve verinin yazılması işlemleri yapılmaktadır.

Hls pipeline optimizasyonu kullanılmadığı durumda döngü içerisindeki fonksiyonlar sıralı olarak çalışmaktadır. Hls optimizasyonu kullanıldığında, döngü içerisindeki işlemler eş zamanlı olarak çalışmaktadır ve veri işlenirken eş zamanlı olarak veri okunabilmekte veya işlenen veri çıkışa yazılabilmektedir. Bu durumda, yeni verinin

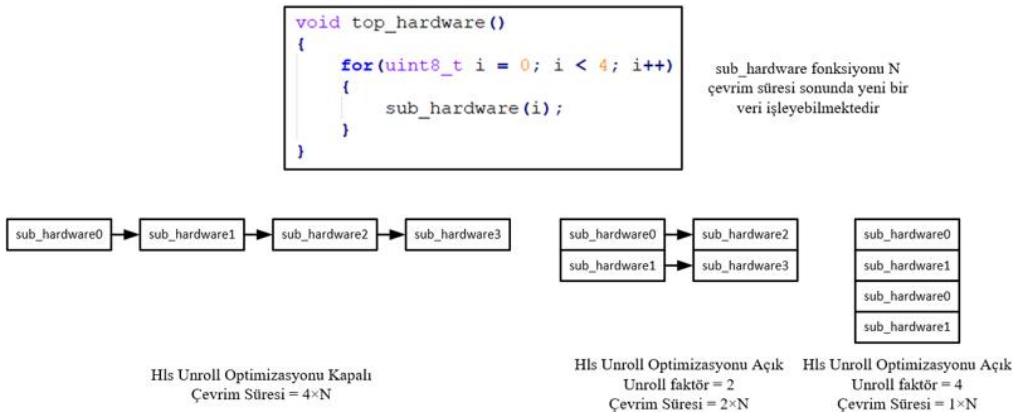
işlenebilmesi için gereken çevrim süresi olan II 1 çevrim olabilmektedir ve bütün döngünün bitirilebilmesi için 4 çevrim süresi yeterli olmaktadır [28].



Şekil 4.31. Hls pipeline optimizasyonu kullanımı çevrim süreleri [28]

4.2.3. Hls unroll optimizasyonu

Vivado HLS ortamında tanımlanan bir döngünün içerisindeki işlemler hls unroll optimizasyonu kullanılmayan tasarımlarda, ayrı çevrim süreleri içerisinde bitirilirler. Tanımlanan döngüde hls unroll optimizasyonu kullanıldığında, döngü içerisindeki fonksiyonun tanımladığı işlemi yapan donanımdan birden fazla sentezlenir bu sayede döngü içerisinde tanımlanan işlem daha hızlı bir şekilde tamamlanır. Hls unroll optimizasyonu kullanıldığında bir unroll faktörü tanımlanmaktadır.



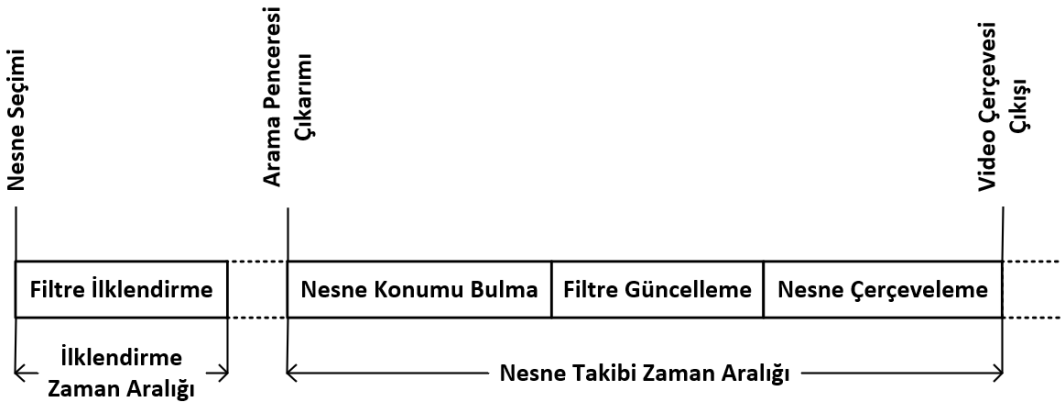
Şekil 4.32. Hls unroll optimizasyonu kullanımı çevrim süreleri

Unroll faktörü, döngü içerisindeki işlemi yapan donanımdan kaç adet sentezleneceğinin tanımlandığı optimizasyon parametresidir. Şekil 4.32'de hls unroll optimizasyonunun kullanımındaki çevrim süreleri gösterilmiştir. Şekilde döngü içerisindeki bir işlemin çevrim süresi N olarak tanımlanmıştır. Hls unroll

optimizasyonu kullanılmadığında, döngü işlemini yapacak bir tane donanım sentezlenecektir ve bütün döngünün işlemini bitirebilmesi için 4×N çevrim süresi beklenecektir. Hls unroll optimizasyonu unroll faktörü 2 olacak şekilde tanımlandığında, döngü içerisindeki işlemi yapacak 2 adet donanım sentezlenecektir ve bütün döngünün işlemini bitirebilmesi için 2×N çevrim süresi beklenecektir. Hls unroll optimizasyonu unroll faktörü 4 olacak şekilde tanımlandığında, döngü içerisindeki işlemi yapacak 4 adet donanım sentezlenecektir ve bütün döngünün işlemini bitirebilmesi için N çevrim süresi beklenecektir. Gösterildiği üzere unroll faktörü sayısının arttırılması sentezlenen donanım sayısını arttırmakta ve çevrim süresini kısaltmaktadır.

4.3. Zamanlama Analizleri

MOSSE algoritması donanım mimarisinde mümkün olduğunda tek AXI4-Stream Video arayüzü kullanılarak, video arayüzleri üzerinde olabilecek zamanlama senkronizasyon problemlerinden kaçınılmaktadır. Her aşamanın kendine özgü AXI4-Stream Video arayüzü ile birlikte çalıştığında, nesne takip zaman aralıkları Şekil 4.33'deki gibi gösterilmektedir.



Şekil 4.33. Nesne takibi zamanlama aralıkları

Nesne takibi zamanlama aralıkları, ilklendirme zaman aralığı ve nesne takibi zaman aralıklarından oluşmaktadır. İlklendirme zaman aralığı, nesnenin ilk seçildiği ve filtrelerin ilklendirilmesi işlemlerinin gerçekleştiği zaman aralığıdır. İlklendirme zaman aralığı, nesne takibi çalışma döngüsünde bir defa çalıştırılmaktadır.

Nesne takibi zaman aralığı, takip edilecek nesnenin seçilmesi ve korelasyon filtresinin ilklendirilmesinin ardından gerçekleştirilen nesne takibi işleminin gerçekleştiği, zaman aralığıdır. İlklendirme aşamasından sonra nesne takibi döngüsel olarak çalışmaktadır. Tasarlanan nesne takip edici FPGA donanımı çalışma hızı performansı analizi yapılırken, döngü halinde çalışan nesne takibi aşamaları ve nesne takibi zaman aralığı dikkatte alınmaktadır.

Nesne takibi zaman aralığı, nesne konumu bulma, filtre güncelleme ve nesne çerçeveleme aşamalarının gerçekleştiği zaman aralıklarından oluşmaktadır. Nesne takibi zamanlama analizi yapılırken, nesne takibi aşamalarının farklı AXI4-Stream Video arayüzleri üzerinden çalışması ve bir nesne takip aşaması gerçekleşirken başka hiçbir aşamanın gerçekleşmemesi durumu dikkate alınmaktadır.

$$t_{ot}=t_a+t_b+t_c \tag{4.6}$$

Nesne takibi zaman aralığı hesaplanırken Denklem (4.6) kullanılmaktadır. Nesne takibi zaman aralığı t_{ot}, nesne konumunun bulunması zaman aralığı t_a, filtre güncelleme zaman aralığı t_b ve nesne çerçeveleme zaman aralığı t_c olarak tanımlanmaktadır.

$$N_{fps}=\frac{1}{t_{ot}} \tag{4.7}$$

Nesne takibi zaman aralığı içerisinde nesne takip edici donanımı saniye başına en yüksek video çerçevesi miktarı, N_{fps} Denklem (4.7) ile hesaplanmaktadır.

Nesne konumu bulma zaman aralığı içerisinde, MOSSE algoritması donanım mimarisindeki, video ön işleme ve korelasyon hesaplama aşamaları gerçekleştirilmektedir. Birbirlerine aynı AXI4-Stream video arayüzü ile bağlı olan Ön işlemci, Çoklu giriş-çoklu çıkış 2D FFT, Korelasyon işlemcisi, 2D IFFT ve Nesne konumu bulucu donanımları, nesne konumu bulma zaman aralığı içerisinde çalıştırılmaktadır. Nesne konumu bulma zaman aralığının analizi için, video ön işleme ve korelasyon hesaplama aşamalarında kullanılan donanımlar video arayüzü ile birbirlerine bağlanmıştır ve oluşturulan donanım Vivado HLS ortamında sentezlenmiştir. Sentezleme sonucunda elde edilen, nesne konumu bulma zaman

aralığında çalıştırılacak donanımların oluşturduğu zamanlama değerlerine, Tablo 4.38'de yer verilmiştir.

Tablo 4.38. Nesne bulma zaman aralığı zamanlama değerleri

Hedef Saat Frekansı	Kestirilen Saat Frekansı (f_{max})	Belirsizlik	Gerekli Saat Sinyali ($tick_{max}$)
100Mhz	107.45Mhz	1Mhz	190855

Tablo 4.38'de elde edilen zamanlama değerleri, Denklem (4.4) ile beraber kullanılarak t_a zamanı değeri hesaplanmaktadır. Bu doğrultuda, nesne konumu bulma zaman aralığında çalışan tüm donanımlar 100Mhz saat frekansı ile çalıştırıldığında, t_a zaman aralığı değeri 1.90855×10^{-3} saniye olarak hesaplanmaktadır.

Filtre güncelleme zaman aralığı içerisinde, MOSSE algoritması donanım mimarisindeki filtre güncelleme aşaması gerçekleştirilmektedir. Birbirlerine aynı AXI4-Stream video arayüzü ile bağlı olan Ön işlemci, Çoklu giriş-çoklu çıkış 2D FFT dönüştürücü ve Filtre güncelleme işlemcisi donanımları, filtre güncelleme zaman aralığı içerisinde çalıştırılmaktadır. Filtre güncelleme zaman aralığının analizi için, filtre güncelleme aşamasında kullanılan donanımlar video arayüzü ile birbirlerine bağlanmıştır ve oluşturulan donanım Vivado HLS ortamında sentezlenmiştir. Sentezleme sonucunda elde edilen filtre güncelleme zaman aralığında çalıştırılacak donanımların oluşturduğu zamanlama değerlerine, Tablo 4.39'de yer verilmiştir.

Tablo 4.39. Filtre güncelleme zaman aralığı zamanlama değerleri

Hedef Saat Frekansı	Kestirilen Saat Frekansı (f_{max})	Belirsizlik	Gerekli Saat Sinyali ($tick_{max}$)
100Mhz	114.285Mhz	1Mhz	262193

Tablo 4.39'de elde edilen zamanlama değerleri Denklem (4.4) ile beraber kullanılarak t_b zaman aralığı değeri hesaplanmaktadır. Filtre güncelleme zaman aralığında çalışan tüm donanımlar 100Mhz saat frekansı ile çalıştırıldığında t_b zaman değeri 2.62193×10^{-3} saniye olarak hesaplanmaktadır.

Nesne çerçeveleme zaman aralığı içerisinde, MOSSE algoritması donanım mimarisindeki nesne çerçeveleme aşaması gerçekleştirilmektedir. Tablo 4.37'de yer verilen zamanlama değerleri Denklem (4.4)'de kullanılarak t_c zaman aralığı değeri

hesaplanmaktadır. 640×480 çözünürlüğündeki bir video girişi ve 100Mhz saat frekansı girişi ile t_c zaman aralığı değeri 3.07204×10^{-3} saniye olarak hesaplanmaktadır.

Hesaplanan t_a, t_b ve t_c zaman aralıkları değerleri, Denklem (4.6)'da kullanılarak t_{ot} zaman aralığı değeri 7.60252×10^{-3} saniye olarak hesaplanmaktadır.

Nesne takip edici donanımının, 640×480 çözünürlüğündeki video girişi, 128×128 arama penceresi boyutu ve 100Mhz saat sinyali girişi değerleri dikkate alınarak t_{ot} zaman aralığı değeri hesaplanmaktadır. Hesaplanan t_{ot} zaman aralığı değeri Denklem (4.7)'de kullanıldığında, N_{fps} değeri @131.53fps hesaplanmaktadır.

4.4. Test ve Similasyon Çalışmaları

Tasarlanan nesne takip edici donanımının doğrulanması için alt modülerin test ve similasyonu ile nesne takip edici donanımı test ve similasyonu olmak üzere iki aşamalı test ve similasyon işlemi yapılmıştır.

Şekil 4.2'deki MOSSE video nesne takip algoritması donanım mimarinde bulunan alt modüllerin test ve similasyon yazılımı, Vivado HLS ortamında tasarlanmıştır. Tasarlanan test ve similasyon yazılımında, alt modüllerin giriş çıkış arayüzleri kullanılarak test verisi verilmiş ve çıkış verisi okunmuştur. Test ve similasyon yazılımı ile elde edilen çıkış verisinin doğrulanması için MATLAB ortamında doğrulama çalışması yapılmıştır. Doğrulama çalışması için, alt modül MATLAB ortamında yazılımsal olarak tasarlanmıştır ve test verisi ile çalıştırılıp çıkış verisi okunmuştur. Her iki ortamda elde edilen test verileri arasındaki benzerlik, Pearson korelasyon katsayısı(p) hesaplanarak incelenmiştir. Pearson korelasyon katsayısı 0 ile 1 arasında bir değerdir. Katsayı değeri 1 değerine yaklaştıkça iki çıkış verisinin benzerliği artmaktadır.

Ön işlemci donanımı, 2D FFT donanımı, Filtre güncelleyici donanımı, 2D Gauss hesaplayıcı donanımı ve Korelasyon hesaplayıcı donanımı doğrulama çalışmaları için Pearson korelasyon katsayısı kullanılmıştır. Elde edilen Pearson korelasyon katsayı değerleri Tablo 4.40'de yer almaktadır.

Tablo 4.40. Alt Modüller Pearson Korelasyon Katsayısı değerleri

Alt Modül	Pearson Korelasyon Katsayısı (p)
Ön İşlemci donanımı	0.9939
2D FFT donanımı	0.9842
2D Gauss Hesaplayıcı Donanımı	0.992
Filtre Güncelleyici Donanımı	1.0
Korelasyon Hesaplayıcı Donanımı	1.0

Elde edilen p katsayısı değerleri 1 değerine yakın çıkmıştır. Bu doğrultuda alt modüllerin Vivado HLS ortamında elde edilen çıktıları ile MATLAB ortamında elde edilen çıktılarının birbirlerine yüksek oranda benzerlik gösterdiği görülmüştür.

Nesne konumu bulucu donanımının doğrulanması için farklı bir doğrulama yöntemi izlenmiştir. MATLAB ortamında farklı noktalarda tepe noktası olan boyutları arama penceresi boyutları ile aynı olan iki boyutlu gauss dağılım matrisi üretilmiştir. Hesaplanan gauss dağılım matrisleri, Vivado HLS ortamında hazırlanan test ve yazılım similasyonu kullanılarak nesne konumu bulucu donanımına giriş arayüzü ile verilmiştir ve çıkış değeri okunmuştur. Okunan değer, MATLAB'de üretilen gauss dağılım matrisi tepe noktası değerleri ile karşılaştırılmıştır. Karşılaştırma sonucunda elde edilen çıkış verileri Tablo 4.41'de verilmiştir.

Tablo 4.41. Nesne konumu bulucu donanımı test ve similasyonu sonuçları

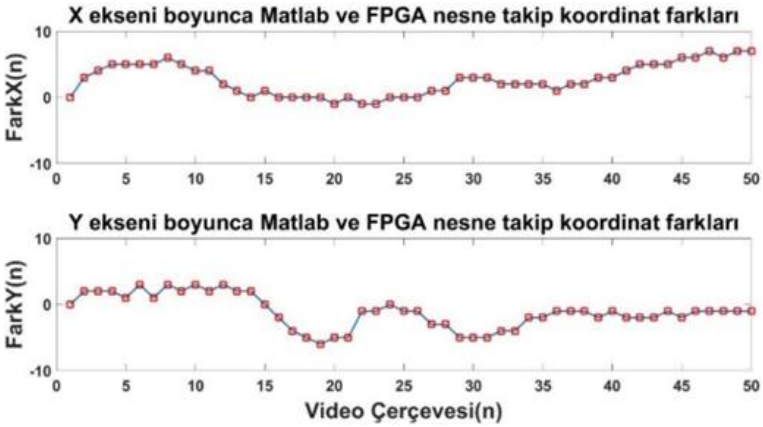
Veri Seti Numarası	Üretilen 2D Gauss Dağılım Matrisi Tepe Noktası (x,y)	Nesne Konumu Bulucu Donanımı Çıktısı (x,y)
1	50,50	50,50
2	72,46	72,46
3	82,16	82,16
4	10,21	10,21
5	37,60	37,60
6	100,105	100,105
7	86,3	86,3
8	7,90	7,90
9	15,110	15,110
10	120,1	120,1

Tablo 4.41'de 10 adet veri seti ile yapılan doğrulama işlemi sonucunda, Nesne konumu bulucu donanımı çıktılarının, MATLAB ortamında hazırlanan gauss dağılım matrisi tepe noktaları ile birebir uyuştuğu görülmüştür ve Nesne konumu bulucu donanımı doğrulanmıştır.

Nesne çerçeveleme işlemcisi donanımının doğrulanması için Vivado HLS ortamında test ve similasyon yazılımı tasarlanmıştır. Tasarlanan test ve similasyon yazılımı kullanılarak Nesne çerçeveleme işlemcisi donanımına video arayüzü ile video çerçevesi gönderilmiştir ve çıkış arayüzü ile çerçevelenmiş video çerçevesi okunmuştur. Çıkış arayüzü ile elde edilen video çerçevesi üzerinde giriş arayüzü ile verilen konumlar doğrultusunda çerçeveleme işlemi yapıldığı gözlemlenmiştir.

Şekil 4.34. Nesne çerçeveleme işlemcisi çıktıları

MOSSE video nesne takip edici donanımı alt modüllerinin, test ve similasyonu yapılarak doğrulanmasının ardından nesne takip edici donanım mimarisi test ve similasyon yazılımı tasarlanmıştır. Nesne takip edici donanım mimarisinin test ve similasyonu için alt modül donanımları ve Şekil 4.2'deki donanım mimarisi kullanılarak Vivado HLS ortamında test ve similasyon yazılımı tasarlanmıştır. Nesne takip edici donanımının performansının doğrulanması için aynı video veri seti, Vivado HLS ortamında tasarlanan test yazılımına ve MATLAB ortamında tasarlanan MOSSE takip edici yazılımına giriş olarak verilmiştir. Elde edilen iki nesne takip koordinat çıktıları karşılaştırılmıştır. Nesne takip edici donanımsal ve yazılımsal olarak gerçeklenmesinin video takip performans karşılaştırılmas Şekil 4.35'de verilmiştir. Yapılan karşılaştırma sonucunda, x ekseni boyunca ±1 , y ekseni boyunca ±10 piksel fark olduğu görülmüştür. Şekil 4.36'da nesne takip edici yazılımsal ve donanımsal gerçeklenmelerine ait aynı video veri seti kullanılarak elde edilen çerçevelenmiş video çıktıları yer almaktadır.



Şekil 4.35. Nesne takip edici performans karşılaştırılması

Şekil 4.36. Nesne takip edici video çıktıları(Kırmızı çerçeve yazılım, sarı çerçeve donanım gerçeklemesi)

5. SONUÇLAR VE ÖNERİLER

Bu tez çalışmasında, MOSSE video nesne takip algoritması yüksek seviye sentez yaklaşımı kullanılarak FPGA donanımı olarak gerçeklenmiştir. Çalışmada literatürde bulunan video nesne takip algoritmaları araştırılmıştır. Yapılan literatür araştırmaları sonucunda, video nesne takip algoritmaları kullandıkları metotlara göre sınıflandırılmıştır. Sınıflandırılan takip algoritmaları incelenerek bir video nesne takip algoritmasından beklenen gereksinimler belirlenmiştir. Çalışmada belirlenen gereksinimleri karşılayabilen MOSSE video nesne takip algoritmasının kullanılması tercih edilmiştir.

Video nesne takip algoritması seçiminin ardından MOSSE algoritmasının matematiksel modellemesi yapılmıştır. Oluşturulan matematiksek modellemeden yola çıkılarak MOSSE algoritması, MATLAB ortamında yazılım tabanlı olarak gerçeklenmiştir. Yazılımsal olarak gerçeklenen MOSSE algoritması, çeşitli video veri setleri kullanılarak test edilmiştir ve nesne takip performansı değerlendirilmiştir.

MOSSE algoritmasının Video nesne takip performansının değerlendirilmesinin ardından donanım mimarisi tasarımı adımına geçilmiştir. Yazılımsal olarak tasarlanan MOSSE algoritması algoritma işlem adımlarına ayrılmıştır ve blok diyagramı oluşturulmuştur. Oluşturulan algoritma blok diyagramından yola çıkılarak FPGA donanım mimarisi tasarlanmıştır.

Donanım mimarisi tasarımında taşınabilirliğin sağlanması için algoritma işlem adımları gerçekleştiren alt donanım modüllerinden oluşturulmuştur. İşlem adımlarını gerçekleştiren alt donanım modülleri aynı türde video veri yolu üzerinden birbirleri arasında veri aktarımı yapabilecek şekilde tasarlanmıştır. Tasarlanan MOSSE algoritması donanım mimarisi Xilinx Zynq SoC mimarisinde çalışacak şekilde tasarlanmıştır. MOSSE algoritmasında bulunan korelasyon filtresinin ilklendirilmesi ve nesne seçimi adımları PS kısmında, yüksek miktarda verinin yüksek hızlı olarak işlendiği kısımlar ise PL kısmında çalıştırılacak şekilde tasarım yapılmıştır.

Önerilen donanım mimarisi tasarım örüntüsü olarak iki adım izlenmiştir. Tasarım örüntüsünün ilk adımı; algoritma alt donanım IP'lerinin FPGA donanım tasarımı ve doğrulanmasının yapılmasıdır. Tasarım örüntüsünün ikici adımı; tasarlanan alt donanım IP'leri kullanılarak üst seviye FPGA donanım mimarisinin tasarlanması ve doğrulanmasının yapılmasıdır.

Alt donanım IP'lerinin FPGA donanım tasarımı ve üst sistem mimarisinin FPGA donanım tasarımı Vivado HLS ortamında yüksek seviye sentez yaklaşımı kullanılarak yapılmıştır. Vivado HLS ortamında yapılan tasarımlarda C++ dili kullanılmıştır. Tasarlanan donanım IP'lerinin oluşturduğu video nesne takip edici FPGA donanımı Xilinx Zynq xc7z030-sbv485-3 SoC'si kullanılarak sentezlenmiştir. Sentezleme sonucunda elde edilen toplam kaynak tüketimi Tablo 5.1'de yer almaktadır.

Tablo 5.1. Video Nesne Takip Edici Donanımı Toplam Kaynak Tüketimi

	BRAM_18K	DSP48E	FF	LUT	URAM
Kullanılan Kaynak Miktarı	148	94	39179	60439	0
Kullanılan Kaynak Yüzdesi	%55.84	%23.5	%24.92	%76.89	%0

Tasarlanan alt donanım IP'lerinin doğrulanması sürecinde Vivado HLS ortamında test ve similasyon yazılımları tasarlanmıştır. Alt donanım IP'lerinin test ve similasyonu için IP giriş arayüzleri ile veri seti verilmiştir ve çıkış arayüzünden çıkış verisi okunmuştur. Elde edilen çıkış verisinin doğrulanması için ilgili donanım IP'sinin MATLAB ortamında yazılımsal tasarımı gerçeklenmiştir ve aynı veri seti verilerek çıkış verisi elde edilmiştir. Elde edilen iki çıkış verisi karşılaştırılarak donanım IP'si doğrulanmıştır.

Donanım IP'lerinin doğrulanmasının ardından önerilen MOSSE algoritması donanım mimarisinin doğrulanması için Vivado HLS ortamında test ve similasyon yazılımı tasarlanmıştır. Tasarlanan similasyon yazılımı ile çeşitli video veri setleri MOSSE algoritması donanım mimarisine verilmiştir ve piksel koordinat düzleminde nesne konumları ve çerçevelenmiş video çıktıları elde edilmiştir. Elde edilen nesne takip çıktılarının doğrulanması için MATLAB ortamında yazılımsal olarak tasarlanan MOSSE algoritmasına aynı video veri seri verilmiştir ve nesne takip çıktıları elde edilmiştir. Elde edilen donanımsal ve yazılımsal gerçeklemelerin nesne takip

koordinatları karşılaştırılarak, doğrulama işlemi yapılmıştır. Karşılaştırma sonucunda yazılımsal ve donanımsal gerçeklemeler arasında benzer takip noktalarının üretildiği görülmüştür.

Tez çalışması sonucunda, 640×480 bir video üzerindeki 128×128 boyutlarındaki bir arama penceresi üzerinde nesne takip edebilen nesne takip edici tasarlanmıştır. Zamanlama analizleri sonucunda nesne takip edici donanımının, 640×480 çözünürlüğündeki bir video üzerinde @131.53fps çerçeve hızı ile nesne takip edebildiği görülmüştür.

Tez çalışması sonucunda, 640×480 çözünürlüğündeki bir video üzerinde 128×128 boyutlarındaki bir arama pencere üzerinde nesne takip edebilen MOSSE algoritması donanım mimarisi tasarlanmıştır. Zamanlama analizleri sonucunda belirtilen video çözünürlüğünde @131.53 fps çerçeve hızı ile nesne takip edebildiği görülmüştür. FPGA donanım tasarımı için yüksek seviye sentezleme yaklaşımının kullanılmasının video nesne takip edici gibi karmaşık görüntü işleme uygulamasının tasarımında zaman ve tasarım maliyetine olumlu etkisi gözlemlenmiştir.

İleriye dönük çalışmalarda, video nesne takip edici donanım mimarisine, nesnenin yaklaşmasına veya uzaklaşmasına adapte olabilmesi için takip işlemi devam ederken arama penceresi boyutlarını dinamik olarak güncelleyebilme özelliğinin kazandırılması hedeflenmektedir. Nesne takip edici tasarımında 2DFFT dönüştürücü donanımının, arama penceresi seçiminde ve FPGA donanımının kapladığı alanda etkisinin önemli olduğu görülmüştür. 2DFFT dönüştürücü donanımının kapladığı alanı azaltmak için bulundurduğu matris tamponunu entegre dışı hafıza elemanlarına taşınmasına yönelik tasarımlarda ileriye dönük çalışmalar arasında hedeflenmektedir.

KAYNAKLAR

[1] Yılmaz A., Javed O., Shah M., Object Tracking: A Survey, *Association for Computing Machinery*, DOI: 10.1145/1177352.1177355.

[2] Haritaoğlu I., Harwood I., Davis L.S., W4: Real-Time Surveillance of People and Their Activities, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, DOI: 10.1109/34.868683.

[3] Ergezer H., Leblebicioğlu H., Visual Detection and Tracking of Moving Objects, 2007 *IEEE 15th Signal Processing and Communications Applications*, DOI: 10.1109/SIU.2007.4298624.

[4] TALU M.F, Nesne takip yöntemlerinin sınıflandırılması, *İstanbul Ticaret Üniversitesi Fen Bilimleri Dergisi*, 2010, **9**(18), 45-63.

[5] Comaniciu D., Ramesh V., Meer P., Kernel-Based Object Tracking, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, DOI: 10.1109/TPAMI.2003.1195991.

[6] Motamed C., Motion detection and tracking using belief indicators for an automatic visual-surveillance system, *Image and Vision Computing*, 2006, **24**(11), 1192-1201.

[7] Zhang K., Zhang L., Liu Q., Zhang D., Yang M. H., Fast Visual Tracking via Dense Spatio-temporal Context Learning, *European Conference on Computer Vision*, DOI: 10.1007/978-3-319-10602-1_9.

[8] Henriques J. F., Caseiro R., Martins P., Batista J., High-Speed Tracking with Kernelized Correlation Filters, *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 2014, **37**(3), 583-596.

[9] Liu T., Wang G., Yang Q., Real-time part-based visual tracking via adaptive correlation filters, *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, DOI: 10.1109/CVPR.2015.7299124.

[10] Bolme D. S., Beveridge J. R., Draper B. A., Lui Y. M., Visual object tracking using adaptive correlation filters, *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, DOI: 10.1109/CVPR.2010.5539960.

[11] Kowalczyk M., Prezewlocka D., Kryjak T., Real-Time Implementation of Adaptive Correlation Filter Tracking for 4K Video Stream in Zynq UltraScale+ MPSoC, *Conference on Design and Architectures for Signal and Image Processing (DASIP)*, DOI:10.1109/DASIP48288.2019.9049203.

[12] Song K., Yuan C., Gao P., Yunxu S., FPGA-based Acceleration System for Visual Tracking, *14th IEEE International Conference on Solid-State and Integrated Circuit Technology (ICSICT)*, DOI: 10.1109/ICSICT.2018.8565781.

[13] Yang H., Yu J., Wang S., Peng X., Design of airborne target tracking accelerator based on KCF, *7th International Symposium on Test Automation and Instrumentation*, DOI: 10.1049/joe.2018.9159.

[14] Gao P., Yuan R., Lin Z., Zhang L., Zhang Y., A Novel Low-cost FPGA-based Real-time Object Tracking System, *IEEE 12th International Conference on ASIC (ASICON)*, DOI: 10.1109/ASICON.2017.8252560.

[15] Liu W., Chen H., Ma L., Moving object detection and tracking based on ZYNQ FPGA and ARM SoC, *IET International Radar Conference 2015*, DOI: 10.1049/cp.2015.1356.

[16] Lu X., Ren D., Yu S., FPGA-based real-time object tracking for mobile robot, 23-25 Kasım 2010, Shanghai, China.

[17] https://vru.vibrationresearch.com/lesson/tables-of-window-function-details/, (Ziyaret tarihi: 17 Nisan 2021).

[18] https://www.xilinx.com/html_docs/xilinx2017_4/sdaccel_doc/odz1504034293215.html, (Ziyaret tarihi: 12 Aralık 2020).

[19] Anuar S., Murad Z., Nazrin M., Isa M., Ismail C., Digital and Analogue Electronics Circuits and Systems, 2015.

[20] Brown S., Rose J., *FPGA and CPLD architectures: a tutorial*, cilt 12, 1996, **13**(2), 45-57, pp. 42-57.

[21] Katz R., Label K., Wang J. J., Cronquist B., Koga R., Pensiz S., Swift G., *Radiation effects on current field programmable technologies*, 1996, **44**(6), 1945-1956.

[22] https://hardwarebee.com/ultimate-guide-fpga-design-flow/., (Ziyaret tarihi: 3 Nisan 2021).

[23] https://www.xilinx.com/products/silicon-devices/soc/zynq-7000.html, (Ziya yaret tarihi: 4 Nisan 2021).

[24] https://lauri.xn--vsandi-pxa.com/hdl/zynq/axi-stream.html, (Ziyaret tarihi: 10 Nisan 2021).

[25] (PG020), AXI Video Direct Memory Access v6.2, 2016, Xilinx.

[26] https://forums.xilinx.com/ t5 / Design-and-Debug-Techniques-Blog/ Video-Series-26-Examples-of-advanced-uses-of-the-AXI-VDMA-IP/ba-p/947566, (Ziyaret tarihi: 17 Nisan 2021).

[27] Fast Fourier Transform v9.1 LogiCORE IP Product Guide PG109, Xilinx, 2021.

[28] https://www.xilinx.com/html_docs/xilinx2019_1/sdaccel_doc/hls-pragmas-okr1504034364623.html#sxx1504034358866, (Ziyaret tarihi: 14 Mayıs 2021).

[29] https://www.xilinx.com/html_docs/xilinx2017_4/sdaccel_doc/odz1504034293215.html, (Ziyaret tarihi: 12 Aralık 2020).

[30] Neuendorffer S., Li T. Wang D., Accelerating OpenCV Applications with Zynq-7000 All Programmable SoC using Vivado HLS Video Libraries, 2015, Xilinx.

KİŞİSEL YAYIN VE ESERLER

[1] **Tunçay E**., Çelebi A., Implementation of MOSSE Object Tracking Algorithm on FPGA with High Level Synthesis Approach, *12th International Conference on Electrical and Electronics Engineering (ELECO)*, Bursa, Turkey, 26-28 November 2020.

ÖZGEÇMİŞ

Cumhuriyet ilköğretim okulunda ilköğretimini ve orta öğretimini tamamladı. Handan Hayrettin Yelkikanat anadolu teknik lisesinde lise eğitimini tamamladı. 2013 yılında mühendislik eğitimi almak üzere Sakarya üniversitesi Mekatronik mühendisliği bölümüne başladı. Elektronik ve gömülü sistemler üzerine daha fazla gelişim sağlamak amacıyla 2015 yılında Elektrik Elektronik Mühendisliği bölümünde çift anadal programına başladı. 2017 yılında Mekatronik mühendisliği bölümünden, 2018 yılında ise Elektrik Elektronik Mühendisliği bölümünden mezun oldu. Lisans eğitimi süresinde ERA Elektronik, ASAŞ Alüminyum ve TOYOTA Otomotiv Sanayi Türkiye A.Ş. gibi kurumlarda uzun dönem stajlar yaptı. 2017 yılında Kocaeli Üniversitesi Elektronik ve Haberleşme Mühendisliği bölümünde yüksek lisans eğitimine başladı. Yüksek lisans çalışmalarının yanında İdealab firmasında kargo taşıyan insansız hava araçları uçuş kontrolü konusunda, FEMSAN elektrik motorları firmasında taktik tekerlekli araçlar için elektro mekanik sistem geliştirme konusunda çalıştı. 2020 yılının Ekim ayında başladığı ASELSAN ve ALTINAY firmalarının iştiraki olan DASAL havacılık firmasında, aviyonik sistemler biriminde gömülü yazılım mühendisi olarak görev almaktadır ve Elektronik ve Haberleşme Mühendisliği ana bilim dalından yüksek lisans derecesinde mezun olma durumundadır.