T.C.
SÜLEYMAN DEMİREL ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

GÖRÜNTÜ İŞLEME VE BEŞ EKSENLİ ROBOT KOL İLE ÜRETİM BANDINDA NESNE DENETİMİ

Fatih Ahmet ŞENEL

Danışman
Doç. Dr. Bayram CETİŞLİ

YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
ISPARTA - 2013

TEZ ONAYI

Fatih Ahmet ŞENEL tarafından hazırlanan **"Görüntü İşleme ve Beş Eksenli Robot Kol ile Üretim Bandında Nesne Denetimi"** adlı tez çalışması aşağıdaki jüri üyeleri önünde Süleyman Demirel Üniversitesi Fen Bilimleri Enstitüsü **Bilgisayar Mühendisliği Anabilim Dalı**'nda **YÜKSEK LİSANS TEZİ** olarak başarı ile savunulmuştur.

Danışman	**Doç. Dr. Bayram CETİŞLİ** Süleyman Demirel Üniversitesi	.....
Jüri Üyesi	**Yrd. Doç. Dr. Ecir Uğur KÜÇÜKSİLLE** Süleyman Demirel Üniversitesi	.....
Jüri Üyesi	**Yrd. Doç. Dr. Mehmet Fatih ÇAĞLAR** Süleyman Demirel Üniversitesi	.....

Enstitü Müdürü	**Prof. Dr. Mehmet Cengiz KAYACAN**	.....

TAAHHÜTNAME

Bu tezin akademik ve etik kurallara uygun olarak yazıldığını ve kullanılan tüm literatür bilgilerinin referans gösterilerek tezde yer aldığını beyan ederim.

Fatih Ahmet ŞENEL

İÇİNDEKİLER

Sayfa

ÖZET

Yüksek Lisans Tezi

GÖRÜNTÜ İŞLEME VE BEŞ EKSENLİ ROBOT KOL İLE ÜRETİM BANDINDA NESNE DENETİMİ

Fatih Ahmet ŞENEL

Süleyman Demirel Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Doç. Dr. Bayram CETİŞLİ

Bu tez çalışmasında gömülü sistem ve görüntü işleme araçlarını bir arada kullanarak robot kollar vasıtasıyla üretim bandı üzerinde hareket eden nesnelerden üretim hatası olanların ayrıştırılması işlemi gerçekleştirilmiştir. Bu işlem için mini6410 ARM işlemcili bir gömülü sistem geliştirme kartı kullanılmıştır. Gömülü sistem, Gömülü Linux işletim sistemi ile çalışmaktadır.

Robot kol olarak beş eksenli hareket edebilen ED-7220C model robot kol ve ED-MK4 robot kol kontrol ünitesi kullanılmıştır. ED-7220C robot kol ED-MK4 kontrol ünitesi üzerinden seri port ile haberleşmektedir.

Uygulama, Linux işletim sistemi yüklü bilgisayarda geliştirilmiş daha sonra çapraz derleme araçları ile ARM işlemci mimarisine göre derlenerek geliştirme kartına yüklenmiştir.

Görüntü işleme aşamaları için açık kaynak kodlu OpenCV kütüphanesi kullanılmıştır. Yazılım geliştirmede OpenCV ile uyumlu olarak çalışan C++ dili ve ücretsiz olarak dağıtımı yapılan Qt C++ geliştirme ortamı kullanılmıştır.

Yapılan çalışma ile bant üzerindeki iki tip nesneden bir türü kameradaki görüntüde tespit edildiğinde, robot kolu ile bant üzerinden alınarak başarıyla hatalı ürün sepetine taşınmıştır.

Anahtar Kelimeler: Gömülü sistem, robot kol, gömülü linux, ED-7220C robot kol, görüntü işleme.

2013, 88 sayfa

ABSTRACT

M.Sc. Thesis

OBJECT CONTROL ON PRODUCTION LINE WITH IMAGE PROCESSING AND FIVE AXIS ROBOT ARM

Fatih Ahmet ŞENEL

Süleyman Demirel University
Institute of Science
Department of Computer Engineering

Supervisor: Doç. Dr. Bayram CETİŞLİ

In this thesis moving objects on production line those have manufacturing defect have been separated by using a combination of embedded systems and image processing via robot arms. For this process, the embedded system development board with mini6410 ARM processor has been used. Embedded system has been operated with Linux operating system.

ED-7220C model robot arm that has moving ability on five axis, and ED-MK4 robot arm control unit have been used. ED-7220C robot arm communicates on ED-MK4 robot control unit with serial port.

The application has been developed on the computer that has linux operation system. Then the application has been installed to development board by compiling with cross compile tools according to ARM processor architecture.

Open source OpenCV library has been used for image processing stages. C++ programming language that is compatible with OpenCV and free distributed Qt C++ development tool have been used for improving software.

With this study when a type of object between two types of object which is on production line is detected in the camera image, it has been carried successfully to defect product basket by taking out of the production line with robot arm.

Keywords: Embedded systems, robotic arm, embedded linux, ED-7220C robot arm, image processing.

2013, 88 pages

TEŞEKKÜR

Bu araştırma için beni yönlendiren, karşılaştığım zorlukları bilgi ve tecrübesi ile aşmamda yardımcı olan değerli danışman hocam Doç. Dr. Bayram CETİŞLİ'ye teşekkürlerimi sunarım.

Araştırmanın yürütülmesinde maddi ve manevi yardımlarını gördüğüm CAD-CAM Araştırma ve Uygulama Merkezi Müdürü Yrd. Doç. Dr. Oğuz ÇOLAK'a ve Arş. Gör. Asım Sinan YÜKSEL'e teşekkür ederim.

3216-YL1-12 No'lu Proje ile tezimi maddi olarak destekleyen Süleyman Demirel Üniversitesi Bilimsel Araştırma Projeleri Yönetim Birimi Başkanlığı'na teşekkür ederim.

Tezimin her aşamasında beni yalnız bırakmayan eşime ve aileme sonsuz sevgi ve saygılarımı sunarım.

Fatih Ahmet ŞENEL
ISPARTA, 2013

ŞEKİLLER DİZİNİ

Sayfa

ÇİZELGELER DİZİNİ

Sayfa

SİMGELER VE KISALTMALAR DİZİNİ

ARM	Acorn RISC machine
CISC	Complex instruction set computing
CMOS	Complementary metal oxide semiconductor
D-H	Denavit – Hartenberg
DNS	Domain name system
DSP	Digital signal processing
FTP	File transfer protocol
GUI	Graphical user interface
HSV	Hue, Saturation, Value
HÜ	Hatalı ürün
HSÜ	Hatasız ürün
MIPS	Million instructions per second
NI	National instruments
PDT	Piksel değerleri toplamı
RISC	Reduced instruction set computing
RGB	Red, green, blue
RTC	Real time clock
SoC	System-on-chip
SIMD	Single instruction multiple data
SPI	Serial Peripheral Interface
SSH	Secure shell
TCP / IP	Transmission control protocol / Internet protocol
UART	Universal asynchronous receiver/transmitter
ÜTİ	Ürün tespit işlemi

1. GİRİŞ

Günümüzün gelişen teknolojik şartlarında, özellikle endüstride insan gücü yerine makine gücü kullanımı hızlı bir şekilde artmaktadır. Hem güvenlik açısından hem zaman açısından makine kullanımının birçok avantajı bulunmaktadır. İnsan gücü yerine makine gücü kullanımı sayesinde ara vermeksizin üretim işlemleri devam etmektedir. İnsan gücünün çalışmasına elverişli olmayan birçok mekânda makine gücü kullanımı fazlasıyla tercih edilmektedir. Öte yandan makinelerin insan gibi yorulma, dikkat dağılması gibi yönleri olmamasından dolayı iş kazaları büyük bir oranda azalmakta ve iş performanslarında herhangi bir düşüş meydana gelmemektedir.

Endüstride en çok kullanılan makineler robotlardır. Robotlar her fabrikanın olmazsa olmazı haline gelmiştir. Her robot bir ana iskeletten ve ana iskelete bağlı kollardan meydana gelmektedir. Robotlar programlanarak, hatasız ve yorulma olmaksızın bir insana göre bin kat daha hızlı çalışabilmektedir. Robot kolların kabiliyetleri meydana geldiği eklem sayıları ile doğru orantılıdır. Ne kadar çok eklemi varsa o kadar hassas işlemler yapabilir. Ancak programlanması da aynı oranda daha zordur. Robot kollar eklemlerden ve her bir eklemi hareket ettiren motorlardan meydana gelmektedir. Motorlar kullanım alanlarına göre farklı türde olabilmektedir. Kontrol edilmeleri ve programlanmaları eklem sayısı arttıkça zorlaştığından robot kollar için motor sürücü devreleri yani kontrol devreleri tasarlanması zorunlu hale gelmiştir. Böylece kontrol işlemi çok daha basite indirgenmiş ve kullanıcıların sadece basit komutlar vererek robot kolları hareket ettirebilmeleri sağlanmıştır.

Hayatımızın hemen her alanına girmiş olan teknolojik cihazlar sadece donanım birimlerinden oluşmaz. Donanım birimlerine akıllı bir şekilde istenilen işleri yaptıracak olan bir de yazılıma ihtiyaç vardır. Akıllı olan her donanım aygıtında, donanımı yönlendiren ve donanımın beyni hükmünde olan bir yazılım mutlaka vardır. Donanımın yazılım ile birleşerek akıllı bir makine haline getirilmesi sonucu elde edilen sistem gömülü sistemler olarak adlandırılmaktadır. Gömülü sistemler çoğunlukla tek bir işi yapmak için tasarlanmış akıllı cihazlardır. Gömülü sistemler basit bir anahtar kontrol devresi olabileceği gibi çok büyük fabrikanın tüm işlemlerini kontrol eden karmaşık yapılarda da olabilir. Gömülü sistemlerin yapacağı

işe göre donanım gereksinimleri vardır. 8 bitlik mikrodenetleyici ile basit işlemler yapabilecek gömülü sistemler tasarlanabileceği gibi çok büyük kontrol gerektiren uygulamalar için 8 bitten daha fazla mimariye sahip mikrodenetleyicilere ihtiyaç duyulacaktır.

Gömülü sistemler için geliştirilmiş 8, 16 ve 32 bitlik mikrodenetleyiciler bulunmaktadır. 32 bitlik mikrodenetleyiciler günümüzde istenebilecek tüm uygulamaları yerine getirebilecek kapasitededir. Fakat tüm uygulamaları 32 bitlik mikrodenetleyiciler ile yapmak gereksiz maliyet getireceğinden mikrodenetleyici seçiminde maliyet ihtiyaç dengesi gözetilmelidir.

Günümüzde bilgisayarlar da dâhil hemen her teknolojik cihazda kullanılan görüntü işleme uygulamaları hızla gelişmektedir. Görüntü işleme bir ortamdan kamera vasıtasıyla alınan görüntülerin, farklı görüntü işleme teknikleri kullanılarak görüntünün analizinin yapılması işlemidir. İstenilen nesneler ön plana çıkarılabilir. Özellikle donanım ile kontrolü yapılan birçok uygulama görüntü işleme teknikleri ile çok daha az maliyetle, bakım gerektirmeksizin gerçekleştirilebilmektedir. Örneğin, sensörler ile kontrol edilen bir üretim bandında ürünlerin istenen noktaya gelip gelmediğini kontrol etmek için en az bir adet sensör gerekmektedir. Gelen ürünlerin düzgün bir şekilde boyanıp boyanmadığını kontrol etmek için farklı bir sensör gerekmektedir. Bu tür bir uygulama, birkaç sensör ile yapılması gereken bir işlem tek bir kamera ile ürünün geldiği tespit edilip renk kontrolü yapılarak çok daha hızlı bir şekilde gerçekleştirilebilir. Sistemlerde kullanılan donanım miktarı ne kadar fazla ise (sensör vs. gibi) sistem o kadar risklidir. Çünkü her bir donanım aygıtının arıza yapma riski vardır ve ne kadar çok donanım aygıtı varsa sistem arıza yapma açısından o kadar risklidir. Oysa görüntü işlemede birçok donanım aygıtı yerine bir ya da birkaç kamera kullanılarak risk oranı bir hayli azaltılabilir.

Görüntü işleme, mobil cihazların artmasıyla birlikte artık sadece masaüstü bilgisayarlarda değil mobil cihazlarda ve gömülü sistemlerde de kullanılmaya başlamıştır. Kullanıcı tarafından cep telefonları ile alınan görüntüler üzerinde işlemler yapmakta, yüz tanıma ile girişlerin gerektiği üst düzey güvenlikli alanlarda görüntü işleme mobil cihazlar ve gömülü sistemler ile gerçekleştirilmektedir. Aksi takdirde çok küçük donanım aygıtları ile yapılabilecek uygulamalarda her kamera

için bir adet bilgisayar gerekecek ve sistemin maliyeti artacaktır. Gömülü sistemler hem maliyet açısından hem de kapladığı alan bakımından bilgisayarlara göre bir hayli avantajlıdır.

Gömülü sistemlerde kullanılan mikrodenetleyiciler, bilgisayarlarda kullanılanlar ile yarış yapabilecek kadar gelişmiştir ve gelişmesini sürdürmektedir. 32 bitlik mikrodenetleyici olan ARM işlemciler üretilmeye başladığı ilk andan beri 32 bitlik bir mimariye sahiptir. Günümüz bilgisayarları 64 bit olarak üretilmektedir. Gömülü sistemler tek bir işi yapmak için özel tasarlandığı ve bilgisayarlarımız bir anda birçok işlem yaptığı (görsellik, müzik çalma, anti-virüs taramaları vs. gibi) dikkate alındığında 32 bit olan ama sadece tek bir işe odaklanan gömülü sistemin, bilgisayarlara göre daha performanslı ve daha istikrarlı çalışacağı anlaşılabilir.

Bu tez çalışmasında 32 bitlik bir mikrodenetleyici olan Samsung firmasının ürettiği S3C6410 ARM11 mikrodenetleyicisi kullanılmıştır. S3C6410 mikrodenetleyicisi için FriendlyARM firması mini6410 adıyla bir geliştirme kartı tasarlamıştır. Mini6410 geliştirme kartının özellikleri sonraki bölümlerde anlatılacaktır. S3C6410 mikrodenetleyicisi 667 MHz'e kadar çalışabilmektedir. Ayrıca S3C6410 mikrodenetleyicisi gömülü sistemler için olan işletim sistemlerini de desteklemektedir. Açık kaynak kodlu olması nedeniyle bu tez çalışmasında Gömülü Linux işletim sistemi tercih edilmiş ve uygulamada kullanılmıştır.

Görüntü işleme uygulamaları için yine açık kaynak kodlu olan OpenCV kütüphanesi gömülü sistemler için derlenmiş ve kullanılmıştır. OpenCV kütüphanesi C ve C++ programlama dili ile yazılmıştır. OpenCV ile en iyi performans, C ve C++ programlama dilleri ile kullanıldığında elde edilmektedir. Bu nedenle C++ dili ile uygulamalar geliştirilmiştir. Gömülü sistem, Linux tabanlı çalışmasından dolayı masaüstü bilgisayarda Ubuntu işletim sistemi ile çalıştırılmıştır. Uygulamalar masaüstü bilgisayarda geliştirilmiş ve çapraz derleme araçları ile ARM işlemci mimarisine göre derlenerek mini6410 geliştirme kartında çalıştırılmıştır.

Bu çalışmada beş eksenli ED-7220C model robot kolu kullanılmıştır. ED-7220C model robot kolu için ED-MK4 sürücü kontrol devresi üretici firma tarafından

geliştirilmiş ve kullanıcılara sunulmuştur. ED-MK4 kontrol devresi, bilgisayar ya da diğer gömülü sistemlerle seri port üzerinden haberleşmektedir.

Bu çalışmada üretim bandına ürünler düşmeden önce yontma işlemine tabi tutulmakta ve çeşitli boyut ve ebatlarda olan tahta parçaları yontularak yuvarlak silindir haline getirilmekte ve kırmızı renge boyanmaktadır. Geliştirilen sistem yuvarlak olmayan ürünleri yani üretim hatası olan ürünleri tespit ederek robot kol ile hatalı ürünleri üretim bandından alarak hatalı ürünler sepetine eklemektedir.

Mini6410 geliştirme kartı ile kameradan görüntü alınarak görüntü işleme teknikleri ile üretim bandı üzerinden geçen ürünler tek tek incelenmiş ve yuvarlak şekilde olmayan ürünler belirlenmiştir. Daha sonra seri port üzerinden robot kola hatalı ürün geliyor bilgisi göndererek hatalı ürünlerin ayıklanmasını işlemi gerçekleştirmiştir.

Bu tezin birinci bölümünde görüntü işleme, gömülü sistemler ve kullanılan robot kolundan kısaca bahsedilmiştir. İkinci bölümde gömülü sistemler ile görüntü işleme ve robot kollar ile alakalı şu ana kadar yapılan çalışmalar sunulmuştur. Bu tezin üçüncü bölümünde kullanılan materyal ve yöntem hakkında bilgiler verilmiştir. Dördüncü bölümde tezin gerçekleştirme aşamaları ele alınmış ve ayrıntılı olarak anlatılmıştır. Beşinci bölümde ise elde edilen sonuçlar ve öneriler sunulmuştur.

2. KAYNAK ÖZETLERİ

Iqbal vd. (2012), ED-7220C robot kolun ileri ve ters kinematik hesaplamalarını yapmışlardır. Çalışmalarında D-H değişkenleri kullanarak homojen dönüşüm yöntemini kullanmışlardır.

Islam vd. (2012), ED-7220C robot kolun kinematik hesaplamalarını yapmışlardır. Ayrıca görüntü işleme kullanarak nesnelerin koordinatlarını belirleyip ters kinematik yöntemi ile nesneyi robot kol ile almayı başarmışlardır.

Huang vd. (2011), bir düzlemde bulunan nesnelerin görüntülerini kameradan aldıktan sonra resimde bulunan nesneleri şekil olarak ayrıştırmışlar ve robot kol kullanarak yuvarlak olan nesneleri başka bir yere yerleştirmişlerdir. Çalışmalarında yatay olarak hareket eden robot kol kullanmışlardır. Kenar belirleme algoritmaları ile nesnelerin kenarları belirlenmiş ve yuvarlak olan nesneler tespit edilmiştir. Robot kolun konumunu yuvarlak olan nesnenin üzerine ayarlanıp bu nesne robot kol ile tutularak başka bir yere konulmuştur.

Wei ve Cai (2010), Gömülü Linux sistemi kullanarak bir ortamın sürekli görüntüsünü alarak hareket tespiti yapmışlardır. Çalışmalarında S3C2440 ARM işlemcili bir elektronik kart kullanmışlardır. Ortamın sürekli olarak görüntülerini almış ve her görüntüyü bir önceki ile karşılaştırarak oluşan farkları incelemiş ve hareketleri tespit etmişlerdir.

Jiao ve Qin (2011), Gömülü Linux kullanarak bir ortamdan alınan görüntüleri gerçek zamanlı olarak internet üzerinden uzak bir bilgisayara aktarmışlardır. Uygulamalarında OMAP3530 ARM işlemcili ve Linux yüklü bir elektronik kart kullanmışlardır. Kameradan aldıkları görüntüyü statik IP'ye sahip olan elektronik karta gerçek zamanlı olarak kaydetmişlerdir. Uzak bilgisayar üzerinden de JAVA ile geliştirdikleri bir arayüz ile statik IP'ye bağlanıp kaydedilen resmi ekrana aktarmışlardır. Böylece gerçek zamanlı olarak görüntüyü aktarmışlardır.

Fanhua vd. (2010), S3C2440 ARM işlemcili, Gömülü Linux işletim sistemi kurarak USB kameradan görüntü almışlardır. Öncelikle ARM işlemcilere Linux işletim

sistemini kurmuşlar, daha sonra USB üzerinden kamera bağlayarak gerçek zamanlı olarak video kaydı yapmışlardır.

Liping ve Kai (2010), yaptıkları çalışmada Gömülü Linux yüklü ARM tabanlı bir elektronik kart üzerinden temel görüntü işleme tekniklerini gerçekleştiren bir yazılım geliştirmişlerdir. Kart belleğinde kayıtlı olan bir resim seçilerek menüde bulunan görüntü işleme seçeneklerinden hangisi seçildi ise o işlem resim üzerinde gerçekleştirilip ekrana sonucu göstermektedir. Görüntü işleme olarak; geometrik dönüşüm, histogram eşitleme, kenar belirleme, şekil bulma, renk dönüşümleri, kenar yumuşatma işlemleri, resim döndürme, yatay ve dikey simetri alma, resim yakınlaştırma işlemlerini gerçekleştirmişlerdir.

Wang vd. (2012), yaptıkları çalışmada Gömülü Linux yüklenmiş S3C2442 ARM işlemci tabanlı elektronik kart ile gerçek zamanlı olarak internet üzerinden görüntü aktarımı yapmışlardır. Bu işlemi yaparken öncelikle USB kameradan aldıkları görüntüleri YUV formatına dönüştürmüşlerdir. Daha sonra BMP olarak kayıt ederek Libjpeg metodu ile sıkıştırmışlardır. Sıkıştırılan görüntü Ftplib teknolojisi ile uzak bilgisayara gönderilmiştir.

Zhang vd. (2012), çalışmalarında S3C210 ARM işlemcili Linux kart ile aldıkları görüntüleri endüstriyel Ethernet hattı ile PC'lere aktarmışlardır. Aynı alt ağ üzerinde bulunan kart ile PC arasında farklı görüntü çözünürlükleri deneyerek 640x480 çözünürlüğünde saniyede 17 ile 22 arasında pencere sayısında görüntü aktarmışlardır.

Seelye vd. (2010), çalışmalarında uzaktan kumandalı robot kol kullanarak, robot kola bağlı olan kamera ile bitkilerin büyümelerini incelemişlerdir. Bitkilerin kas gelişimlerini ayrıntılı olarak incelemişlerdir. Bitki büyüdükçe kamera açısını robot kol ile ayarlayarak istenilen bölgelerin gelişimini takip etmişlerdir. Uzaktan kumanda ederken veri haberleşmesini ZigBee haberleşme teknolojisini kullanarak gerçekleştirmişlerdir. Kullandıkları robot kol servo motor ile tasarlanmıştır. Sony FCB-IX11AP model renkli kamera kullanmışlardır ve kameradan aldıkları görüntüleri de ZigBee teknolojisi ile aktarmışlardır.

Liu ve Chou (2009), yaptıkları çalışmada CMOS OV9650 görüntü sensörü ve Gömülü Linux yüklü bir PXA270 işlemcili ARM tabanlı elektronik kart kullanmışlardır. Sensörden aldıkları görüntü verilerini I^2C haberleşme protokolü ile ARM işlemcisine aktarmışladır. Geliştirdikleri sistem düşük güç tüketimi ile ön plana çıkmakta ve mobil robot uygulamalarında kullanılmaya elverişli yapıdadır. Geliştirdikleri sistem VGA formatında saniyede 15, QVGA formatında saniyede 30 pencere alabilmektedir. Çalışmalarında 320x240 çözünürlükte görüntüler almış ve ARM işlemcili karta iletmişlerdir.

Rapp (2011), yaptığı çalışmada robot kola bağlı bir raket üzerinde pin pon topunu takip eden ve sektiren bir sistem tasarlamıştır. Bu sistemde iki adet kamera bulunmaktadır, kameralar Linux tabanlı çalışan bir bilgisayara bağlıdır. Bilgisayar bir adet endüstriyel bilgisayara bağlıdır. Robot kol endüstriyel bilgisayar tarafından kontrol edilmektedir. Kameralardan biri raketi üstten, diğeri yandan görecek şekilde yerleştirilmiştir. Böylece topun yukarı doğru yükselmesi ile raketin vurma hızını, sağa sola kayması ile de raketin konumunu ayarlamıştır. Bu sistem yaklaşık olarak 30 dakika boyunca topu sektirmeye devam edebilmektedir.

Anh ve Song (2010), yaptıkları çalışmada nesneleri yakalayabilen bir sistem geliştirmişlerdir. Robot kol üzerine bağlı kamera ile yerde hareket halinde olan nesneleri takip etmekte ve robot kol servo motorlar ile kontrol edilmektedir. İlk olarak görüntü alınmakta ve daha sonra alınan görüntüler önceki görüntüler ile karşılaştırılmaktadır. Hareket algılandığı zaman hareket halinde olan nesnenin konumu tespit edilmektedir. Robot kolu hareket ettirmek için servo motor sürücü devresine gerekli komutlar gönderilerek nesneye yakalayıp ve tutmaktadır.

Chang ve Shih (2008), yaptıkları çalışmada üç eksenli bir robot kola bağlı silah ile hedef takibi yapan ve hedefi şaşırmadan vurabilen bir sistem tasarlamışlardır. Robot kolun sürücü devresi için NI PXI-7356 hareket kontrol platformunu kullanmışlardır. Hedef tespit edildikten sonra namlunun hedefe en kısa zamanda ulaşabilmesi için hesaplamalar yapmışlar ve bulanık mantık algoritmaları kullanmışlardır. Robot kolu servo motorlar ile kontrol edilmiştir.

Ogawa vd. (2011), yaptıkları çalışmada gerçek zamanlı olarak bir insan ile hava hokeyi oynayabilen robot kol gerçekleştirmişlerdir. Masayı üstten gören bir kamera ile bilgisayara alınan görüntüler işlenmekte ve hokeyin yeri tespit edilmektedir. Daha sonra robot kol hokeyi karşılayacak şekilde konumlandırılmaktadır ve karşılık vermektedir. İki eksenli olan robot kolu hareket ettirmek için kontrol birimi tarafından iki adet açı ve hız bilgisi robot kola gönderilmektedir.

Bayrak ve Sarıtaş (2008), yaptıkları çalışmada beş eksenli bir robot kolu ile hedef tespit edilmesi işlemini gerçekleştirmişlerdir. Bu işlemi yaparken robot kolun karşılaşabileceği engellere çarpmasını önlemişlerdir. Önceden belirlenen hedefler bilgisayara kayıt edilmiş ve ortamdan alınan görüntüde hedef tarama işlemini gerçekleştirmişlerdir. Hedef tespit edildikten sonra koordinat tespiti algoritmaları ile görüntü üzerinde hedefin koordinatları belirlenmiş ve robot kol ilgili koordinata konumlandırılarak hedef alınmıştır.

Basile vd. (2012), yaptıkları çalışmada, birden fazla robot kolu hedef nesnelerin koordinatları hesaplanarak, eş zamanlı olarak çalıştırmışlardır. Bu işlem için hedef nesnenin koordinatları bulunduktan sonra ters kinematik hesabı ile çalışma alanı içinde olan robot kollardan en yakın olanı nesnenin alımı için harekete geçmektedir.

Bejo vd. (2009), bilgisayar destekli bir mikroişlemci ile 6 eksenli robot kol için bir kumanda devresi geliştirmişlerdir. Aynı zamanda ters kinematik hesaplamalar yaparak, bilgisayar üzerinden kontrolü gerçekleştirmişlerdir.

Luo vd. (2008), ALSOK Guard ROBOi insansı robot kullanarak güvenlik uygulamaları gerçekleştirmişlerdir. Robotun gözlerinde bulunan kameralar vasıtasıyla ortamın görüntüsü alınmış, müdahale edilmesi gereken nesneler tespit edilerek ters kinematik yöntemi ile müdahale edilmiştir. Toplamda 14 adet servo motor bulunan robot üzerinde mesafe ölçüm işlemleri için sensörler bulunmaktadır. Hedef nesneler eğer robotun kollarının çalışma aralığı dışında ise sensörler vasıtasıyla yakınlık ve uzaklık değiştirilerek nesneleri almayı başarmışlardır.

Low (2010), yaptığı çalışmada web tabanlı kamera üzerinden aldığı görüntüleri kablosuz üzerinden uzak bilgisayara göndermiştir. Uzak bilgisayarda çalışan görüntü

işleme algoritmaları ile gelen görüntülerin kenarlarını belirlemiştir. Yaptığı sistemde iki adet kızılötesi mesafe sensörü kullanmıştır. Yakın olan nesneler tespit edildikten sonra wireless özelliği olan web cam ile görüntüleri kablosuz olarak bilgisayara iletmiştir.

Aby vd. (2011), yaptıkları çalışmada ARM işlemcili bir gömülü sistem kullanarak insan yüzü bulma sistemi gerçekleştirmişlerdir. Görüntü işleme uygulamaları için OpenCV kütüphanesini kullanmışlardır. Sonuç olarak kameradan aldıkları görüntüde bulanan yüzleri kırmızı bir halka ile göstermeyi başarmışlardır.

Deepthi ve Sankaraiah (2011), yaptıkları çalışmada Nokia firmasına ait bir mobil cihaz üzerinde Qt ve OpenCV ile birlikte görüntü işleme uygulaması gerçekleştirmişlerdir. Kameradan aldıkları görüntüyü, gri seviyeye dönüştürmüş, Gaussian Filtre ile yumuşatma işlemi yapmışlardır. Daha sonra histogram eşitleme yaparak kenar çıkarımı yapmışlardır. En son resimde istenilen kısmı kırparak mobil cihazın belleğine resim olarak kaydetmişlerdir.

Ali vd. (2012), yaptıkları çalışmada bilgisayara bağlı bir kameradan aldıkları görüntüde bulunan nesneleri şekil olarak incelemişlerdir. Yuvarlak, kare, dikdörtgen, kalp şeklinde olan nesneleri tanımışlar ve bilgisayara bağlı bulunan robot kol ile istedikleri nesneleri alıp başka bir yere yerleştirmişlerdir.

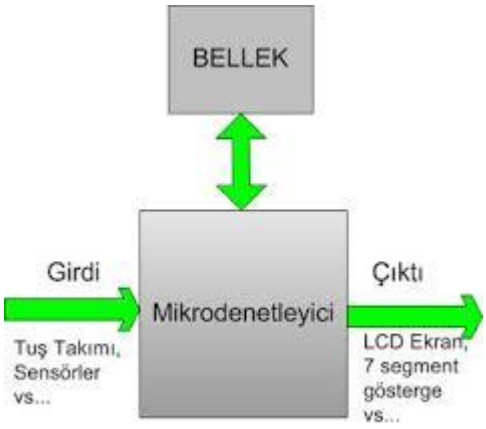
Zheng vd. (2009), çalışmalarında gerçek zamanlı olarak ARM tabanlı bir gömülü sistemle, kameradan görüntü alıp hareketli nesnelerin tespitini yapmışlardır. Arka plan görüntüsünden, alınan kamera görüntüleri çıkarılarak hareketler algılanmıştır. Algıladıkları hareketleri TCP / IP protokolü ile internet üzerinden göndermişlerdir. Karşı tarafta çalışan bilgisayarda alınan görüntüler internet explorer ya da firefox ile gerçek zamanlı olarak gösterilmiştir.

3. MATERYAL VE YÖNTEM

Görüntü işleme teknikleri, yapılan çalışmalarda çoğunlukla MATLAB görüntü işleme kütüphanesi ve OpenCV kütüphaneleri kullanılarak gerçekleştirilmektedir. MATLAB programının ücretli olması geliştirilen uygulamalara ek bir maliyet getirmektedir. Ayrıca, MATLAB kullanılarak geliştirilen uygulamaların çalışması için hedef cihazda MATLAB için gerekli programların yüklenmiş olması gerekmektedir. Gömülü sistemler kısıtlı işlemci ve belleğe sahip olduklarından MATLAB programı ideal bir çözüm değildir. Gömülü sistemler için MATLAB uygun bir çözüm olmadığından açık kaynak kodlu, ücretsiz olarak dağıtılıp geliştirmeye açık olan OpenCV görüntü işleme kütüphaneleri tercih edilmiştir. OpenCV birçok programlama dilinde çalışmayı desteklemektedir. Gömülü sistemler çoğunlukla doğrudan donanıma erişim desteği bulunduğundan dolayı C ve C++ dilleri ile programlanmaktadır. Bu tez çalışmasında C++ programşama dili ve OpenCV görüntü işleme kütüphaneleri kullanılmıştır.

3.1. Gömülü Sistem

Gömülü sistem; tek başına ya da herhangi bir sistemin içine yerleştirilerek çalışan ve dâhil olduğu sisteme akıllı olma özelliği kazandıran donanım ve yazılımın tamamına verilen isimdir. Gömülü sistemlerde bulunan yazılımlar bilgisayarlarda olanlardan farklı olarak kullanıcı ile doğrudan ilişkisi olmayan ve çoğunlukla tek amaç için geliştirilmiş yazılımlardır. Gömülü sistemler tek bir amaca yönelik olduğu için donanımsal parçaları en uygun olacak şekilde minimum gereksinimlerde tasarlanması gerekmektedir. Aynı zamanda gömülü sistemler bir defaya mahsus olmak üzere geliştirilir ve daha sonra seri üretim ile hızlı bir şekilde binlerce üretilerek maliyetler en aza indirgenir. Gömülü sistemlerde bulunan yazılımlar kullanıcıların isteği doğrultusunda değiştirilemezler. Ancak üretici firma tarafından yeni versiyonları çıktıkça güncelleme işlemi ile değiştirilebilirler. Bu türden yazılımlara yarı kalıcı yazılımlar denilmektedir. Yarı kalıcı yazılımlar firmware adı ile anılmaktadır. Genel olarak her gömülü sistem Şekil 3.1'de gösterilen birimlere sahiptir:



Şekil 3.1. Gömülü sistem yapısı

3.1.2. Gömülü linux

Gömülü sistemler genellikle tek bir işi yapmak için tasarlanmış küçük sistemlerdir. Ön plana çıkan özellikleri ise küçük boyutta olmaları, az enerji harcamaları, donanım birimlerinin az olması ve gerektiğinde arttırılabilir olmalarıdır. Gömülü sistemlerin vazgeçilmez özelliği ise tek bir işi yapmak için tasarlanmış olmalarıdır. Gömülü sistemlerin çekirdeğini bir ya da daha fazla mikroişlemci ya da mikrodenetleyici oluşturur. İşletim sistemi kullanan gömülü sistemler Windows tabanlı ya da Linux tabanlıdır. Windows tabanlı olan gömülü sistemler genellikle Windows CE, Embedded Compact 7 tabanlı çalışan sistemlerdir. Linux tabanlı sistemler açık kaynak kodlu olduğu için kullanıcılar çekirdek derleyerek istenilen özellikte ve boyutta probleme özel gömülü Linux sistemi tasarlayabilirler. Bu tez çalışmasında kullanılan gömülü sistem Linux tabanlıdır. Linux seçimindeki en önemli faktör, çekirdeği kullanıcı derlediği için istenilen özellikler çekirdeğe eklenip gerekli olmayan tüm özellikler derleme işlemine dâhil edilemeyebilir ve böylece sistemin daha hızlı çalışmasını ve daha küçük boyutlarda tasarlanmasına imkân sağlamaktadır.

3.1.3. Çapraz derleme (Cross compile) ve zincir aracı (toolchain)

Derleme işlemlerine geçmeden önce çapraz derleme ve zincir aracı ifadeleri açıklanacaktır. Çapraz derleme işlemi bir işlemci üzerinde başka bir işlemci için çalışabilecek dosyaların derlenip oluşturulmasıdır. X86 tabanlı bir sistemde çapraz

derleme işlemi için ARM mimarisine göre kodlar derlenebilir. Çapraz derleme işlemi gömülü sistem geliştiricileri tarafından yaygın bir şekilde kullanılmaktadır. Çünkü çalışma frekansları masaüstü bilgisayarlara göre çok düşük olan gömülü sistem mikrodenetleyicileri ile günlerce sürebilecek olan derleme işlemleri, masaüstü bilgisayarlar ile saniyeler içinde bitirilebilir. Bu işlem sistem geliştiricileri için hız açısından çok önemlidir. Hemen derleme işlemi gerçekleştirilip, gömülü sisteme yüklenerek sonuçlar değerlendirilebilir. Çapraz derleme işlemini gerçekleştirebilmek için gerekli derleme araçlarına zincir aracı denir. ARM işlemciler için internet üzerinden ücretsiz olarak birçok toolchain edinilebilir. Bu tez çalışmasında "arm-linux-gcc-4.5.1" zincir aracı kullanılmıştır. "arm-linux-gcc-4.5.1" toolchain aracının uygulama geliştirilecek bilgisayara nasıl entegre edildiği Ek A'da ayrıntılı bir şekilde anlatılmıştır.

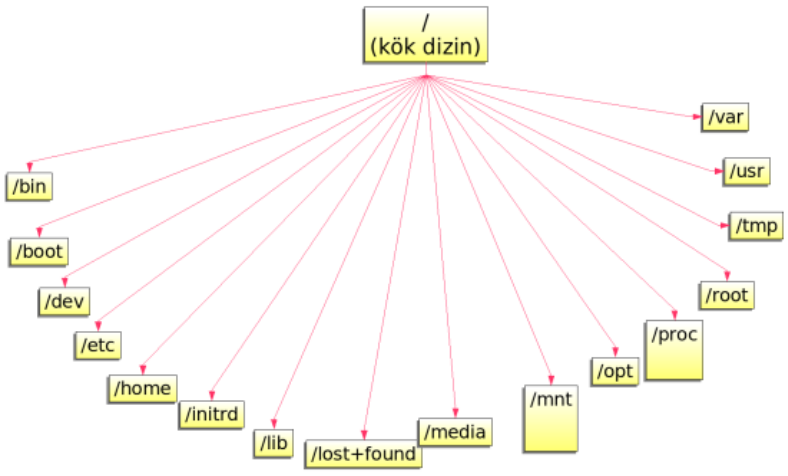
3.1.4. Kernel derleme

Linux çekirdeği; donanım ile yazılım arasında haberleşmeyi sağlayan bir ara katmandır. Kullanılan sistemde ihtiyaç olmayan donanım desteklerinin çekirdeğe eklenmemesi ve ihtiyaç olan donanım desteklerinin çekirdeğe eklenmesi için çekirdek derleme işlemi yapılmaktadır. Bu açıdan bakıldığında Linux işletim sistemleri kullanıcıya birçok noktadan esneklik sağlamaktadır. Gömülü sistemlerde işlemci performansı çok önemli olduğu için, çekirdek derleme işlemi gömülü sistemlerde en önemli aşamalardan birisi olarak kabul edilmektedir. Çünkü işlemcinin performansı; yaptığı işlemler ve kontrol ettiği donanım birimleri ile doğrudan ilgilidir. Bu nedenle gömülü sistem tasarlanırken hazır çekirdekler yerine sisteme uygun olan optimum çekirdek derlenmelidir. Böylece gereksiz hiçbir donanım desteği çekirdeğe dâhil edilmeyecek, çekirdek boyutu küçülecek ve sistem daha hızlı ve daha kararlı çalışacaktır. Bu tez çalışmasında kullanılan gömülü sistem kartı olan mini6410 geliştirme kiti, Linux'un 2.x versiyonlarına destek vermektedir. Linux çekirdeğinin 3.x olan versiyonları henüz bu kart tarafından desteklenmemektadir. Destek verememesinin nedeni, S3C6410 mikrodenetleyicisi için hazır konfigürasyon ayar dosyasının olmamasıdır. Bu dosya kullanıcı tarafından oluşturulabilir, ancak binlerce seçenek arasından S3C6410 mikroişlemcisi için en uygun ayarların seçilebilmesi çok uzun bir süreç alacağından konfigürasyon dosyasının hazır olarak kullanılması en pratik yoldur. En çok kullanım alanına sahip

mikroişlemciler için konfigürasyon dosyaları Linux çekirdek geliştiricileri tarafından hazırlanmakta ve kullanıcılara ücretsiz olarak sunulmaktadır. Bu çalışmada konfigürasyon dosyalarının içinde hazır olarak bulunduğu 2.x versiyonları tercih edilmiştir. Bu çalışmada mini6410 geliştirme kartı ile 7''lik dokunmatik ekran kullanılmıştır. Linux çekirdeği olarak mini6410 7'' ekran için tüm konfigürasyon ayarlarının içinde bulunduğu "config_mini6410_a70" dosyasına sahip Linux-2.6.28 versiyonu kullanılmıştır. Linux çekirdeklerinin en güncel olanı ve istenilirse önceki sürümleri de www.linux.org sitesinden indirilebilir. Bu tez çalışmasında Linux tabanlı gömülü sistem kullanıldığı için masaüstü çalışma bilgisayarı olarak Linux tabanlı Ubuntu 12.10 versiyonu yüklü bir sistem kullanılmıştır. Tez boyunca yapılan tüm çalışmalar bu bilgisayarda gerçekleştirilmiştir. Linux çekirdeğinin derlenmesi, istenilen ayarların yapılaması işlemleri Ek B'de anlatılmıştır.

3.1.5. Linux dizin ve dosya yapısı

Gömülü Linux işletim sisteminin çalışması için her Linux dağıtımında olması gereken dosyalar vardır. Linux sistemlerinde sistem ve kullanıcı ile ilgili her şey dosyalarda tutulmaktadır. Linux sistemlerde kök dizin, "/ (root)" dizinden başlar ve diğer tüm dizinler bu dizinin bir alt dizini olacak şekilde yerleştirilmektedir. Sistem ilk açıldığında kök dizine bakılır, eğer kök dizin mevcut değilse sistem hata verir ve açılma işlemini durdurur. Şekil 3.2'de Linux dosya yapısı görülmektedir.



Şekil 3.2. Linux dizin yapısı

Linux sistemlerde Şekil 3.2'de görülen tüm dizinler olmak zorunda değildir, ancak olmazsa olmaz olan dizinler vardır. Bu olması gerekli olan dizinler kısaca aşağıda anlatılacaktır.

/bin: Terminal penceresinde kullanılan tüm komutlar bu dizin altındadır. /bin dizini varsayılan olarak sistem path'ine ekli durumdadır. Komutlar başka bir dizine kopyalanıp sistem path'ine eklenerek de kullanılabilir ancak varsayılan olarak /bin dizini altına yerleşik durumdadır.

/boot: Boot işlemi, işletim sisteminin yüklenmeye başladığı evredir. /boot dizini boot işlemi için gerekli tüm dosya ve ayarların tutulduğu dizindir. /boot dizininde herhangi bir dosyanın eksik olması sistemin sağlıklı bir şekilde başlatılmasını etkileyecektir.

/dev: Linux sistemlerde her şeyin dosyalarda tutulduğu yukarıda bahsedilmişti. Donanım aygıtları da dosyalarda tutulmaktadır. /dev dizini işletim sisteminde bulunan donanımlar ile ilgili bilgilerin tutulduğu dizindir.

/etc: Linux sisteminin sinir merkezini oluşturmaktadır. Sistem ile ilgili tüm ayarlar bu dizin altında tutulmaktadır.

/home: Kullanıcı dosyalarının tutulduğu dizindir. Her bir kullanıcıya kendi adı ile /home dizini altında bir dizin oluşturulur ve tüm dosyalarını bu dizine oluşturulur.

/initrd: Başlangıç bellek diski olarak adlandırılır. Kernel yüklenme işlemi tamamlandığında kök dizinin üzerine kurulduğu dizindir.

/lib: Sistem ile ilgili ve kullanıcı programları ile ilgili tüm kütüphane dosyalarının tutulduğu dizindir.

/lost+found: Sistem ile ilgili olarak herhangi bir problem meydana gelirse, sistem hasarlı olan dosyaları bu dizin altına kopyalar. Böylece problemli olan dosyalar bu dizine bakılarak tespit edilir.

/media: USB bellek, CD ROM gibi aygıtlar bu dizine bağlanır ve bu dizin ile erişilir.

/mnt: /media dizini gibi kullanılmaktadır.

/opt: Linux dağıtımlarında olmayan standart dışı programlar yüklendiğinde /opt dizini altına yüklenir.

/proc: Sanal bir dosya sistemidir. Sistem ile ilgili olarak ayrıntılı bilgilerin görüntülenebileceği dizindir.

/root: Sistem yönetici bilgilerinin tutulduğu dizindir.

/usr: Linux sistemler ile ilgili tüm programların yüklendiği dizindir. Ayrıca kurulan programlar ile ilgili tüm kütüphane dosyaları /usr/lib dizini altında tutulur.

/var: Sistem ile ilgili değişken bilgilerin tutulduğu dizindir. Yazıcı kuyruğu, sistem Log'ları gibi bilgiler bu dizin altında tutulur.

/tmp: Programlar ile geçici bilgilerin tutulduğu dizindir. Genellikle sistem ilk başladığında bu dizin altında bulunan dosyalar silinerek /tmp dizini temizlenir.

Linux sistemlerde dosya ve dizin yapısı çok önemlidir. Bu nedenle dosya ya da dizin üzerinde işlemler yapılırken çok dikkat edilmesi gerekmektedir. Varsayılan olarak önemli dosya ve dizinlerin izinleri sadece sistem yöneticisi kullanabilecek şekilde değiştirilmiştir. Bu nedenle herhangi sebeple dosya, dizin izinlerinde değişiklik yapmak gerektiğinde tekrar eski haline getirilmelidir.

Gömülü sistemler de Linux işletim sistemi kullanmak için, gömülü sisteme Linux kök dosya sistemi kurulması gerekmektedir. Gömülü sistemler için hazır olarak üretici firmaları tarafından kök dosya sistemleri kullanıcıya sunulmaktadır. Ancak kullanıcı isterse kendisi de kök dosya sistemini oluşturabilmektedir. Bu tez çalışmasında mini6410 geliştirme kartı için baştan bir kök dosya sistemi oluşturulmuştur. Mini6410 geliştirme kartı için Linux kök dosya sistemi kurulumu Ek C'de anlatılmıştır.

3.1.6. FTP, SSH kurulumu ve dosya transferi

FTP, internete bağlı olan iki farklı cihaz arasında dosya transferi yapmaya izin veren dosya transfer protokolüdür. Mini6410 geliştirme kartı ile yazılım geliştirilen masaüstü bilgisayar arasında dosya transferi yapılması gerekmektedir. Aksi takdirde kod üzerinde yapılan her işlem, masaüstü bilgisayarda derlendikten sonra mini6410 geliştirme kartına SD kart üzerinden kopyalanacak ve çok fazla zaman kaybına neden olacaktır. Zaman kayıplarını önlemek için internet üzerinden FTP protokolü kullanarak derlenen her kod saniyeler içinde mini6410'a kopyalanacak ve sonuçlar anında görülebilecektir. Mini6410 geliştirme kartı için hazırlanan SD karta "rootfs/" dizini kopyalanıp kart boot edildiğinde sistem SD kart üzerinden boot ederek açılacaktır. Kullanıcı adı ve parolası soracak, ardından giriş yapılacaktır. Otomatik giriş için gerekli adımlar sonraki bölümde anlatılacaktır. FTP kurulumuna geçmeden önce mini6410'un internet bağlantısının çalışıp çalışmadığını kontrol etmemiz gerekmektedir. İnternet kablosunu takıp mini6410'u başlattıktan sonra komut satırına "ifconfig" komutu yazılarak aktif olan bağlantıları listeleyebiliriz. "eth0" IP adresi almışsa internet bağlantısı çalışıyor demektir. Eğer IP adresi almamışsa "eth0" bağlantı noktasını "ifup eth0" komutu ile aktif etmemiz gerekmektedir. IP adresi aldıktan sonra her hangi bir siteye ping atılarak bağlantı test edilebilir. Eğer site adıyla ping atma işlemi başarısız olursa DNS ayarlarında problem olduğu anlaşılır. Bu sorunu çözmek için site adı yerine IP adresi yazılarak ping atılır. Sonuç başarılı olursa "/etc/apt/sources.list" dosyasında "http://ftp.us.debian.org" adresi yerine, bu web adresinin IP adresini yazarak bu sorun çözülebilir. Sorun çözülmezse internet bağlantısı kontrol edilerek bu işlemler tekrarlanır.

Linux tabanlı sistemlere FTP ve SSH kurulum işlemleri Ek D'de ayrıntılı olarak anlatılmıştır.

3.1.7. Temel linux komutları ve program çalıştırma

Linux tabanlı işletim sistemlerinde her işlem terminal penceresinde komutlar vasıtasıyla gerçekleştirilir. Her ne kadar Ubuntu, Mint, Pardus gibi görsel işletim sistemleri geliştiriliyor olsa da birçok işlemde terminal üzerinden işlem yapmak

kaçınılmazdır. Linux sistemlerde en çok kullanılan komutlar Ek E'de açıklamaları ile birlikte verilmiştir.

3.2. ARM İşlemciler

Gömülü sistemlerde 8, 16 ve 32 bitlik mikrodenetleyiciler kullanılmaktadır. Bu tez çalışmasında gömülü Linux işletim sistemi çalıştırıldığından 32 bitlik bir mikrodenetleyici kullanılması gerekmiştir. 32 bit mikrodenetleyici üreten birçok firma bulunmaktadır. Ancak teknolojide en fazla kullanılan ARM işlemcilerdir. Bu nedenle ARM işlemciler en fazla destek alınabilen işlemcilerdir. Kullanım alanının yaygınlığı sebebiyle tez çalışmasında ARM işlemci tercih edilmiştir.

Tez çalışmasında ARM11 çekirdeğine sahip S3C6410 ARM işlemcisi kullanılmıştır. Geliştirilen sistem dokunmatik ekran ile çalışmaktadır. Aynı zamanda USB bağlantı noktası, Ethernet girişi gerektirmektedir. Tüm bu gereksinimleri karşılayabilen en az maliyetli geliştirme kartı olarak FriendlyARM firmasının ürettiği mini6410 geliştirme kartı tercih edilmiştir. Mini6410 geliştirme kartı üzerinde Samsung firması tarafından üretilen ARM11 çekirdeğine sahip S3C6410 ARM işlemcisi bulunmaktadır. Devam eden başlıklarda kısaca ARM işlemciler ve ARM11 çekirdeği hakkında bilgiler verilecektir.

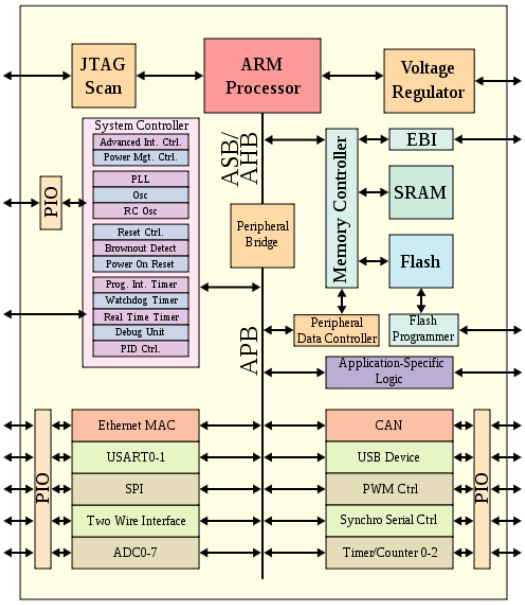
3.2.1. ARM hakkında genel bilgiler

ARM, İngilizce "Advanced RISC (Reduced Instruction Set Computer) Machines" in kısaltmasıdır. RISC işlemci mimarisi ile geliştirilen ve dünyanın en önemli gömülü sistem işlemcilerinden biridir. ARM mimarisi ile geliştirilen ürünlerden bazıları:

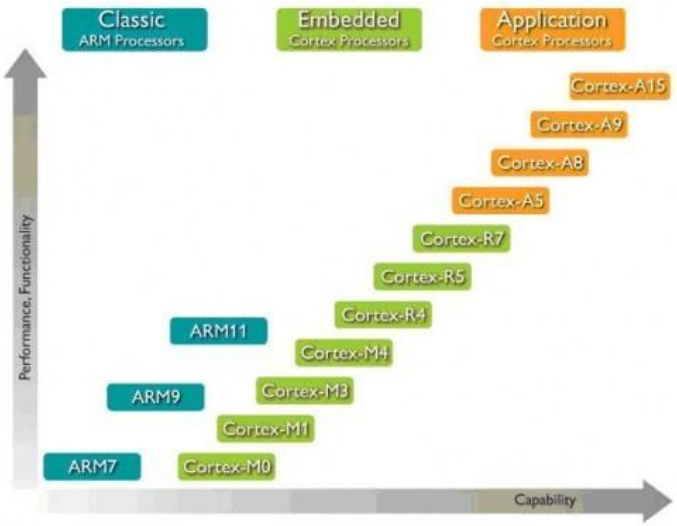
- 16/32 bir RISC işlemciler
- Veri Makineleri (Data Engines)
- 3D İşlemciler
- Gömülü Bellekler
- Çevre Birimleri
- Yazılımlar ve geliştirme araçları olarak sıralanabilir.

ARM mimarisi, dünyanın en yaygın 32 bit mikroişlemcisine sahip mimaridir ve modern SoC (System on Chip) model tasarımlarının en önemli örneğidir. Şekil

3.3.'te ARM işlemcilerin iç yapısı ve Şekil 3.4'te ARM işlemcilerin gelişim evresi görülebilir.



Şekil 3.3. ARM işlemci iç yapısı (Özkan, 2007)



Şekil 3.4. ARM mimarisi gelişim evresi (Özkan, 2007)

ARM, gömülü sistem ve mobil platformda en fazla tercih edilen işlemci mimarisi olduğundan dolayı çok fazla yazılım desteği almaktadır.

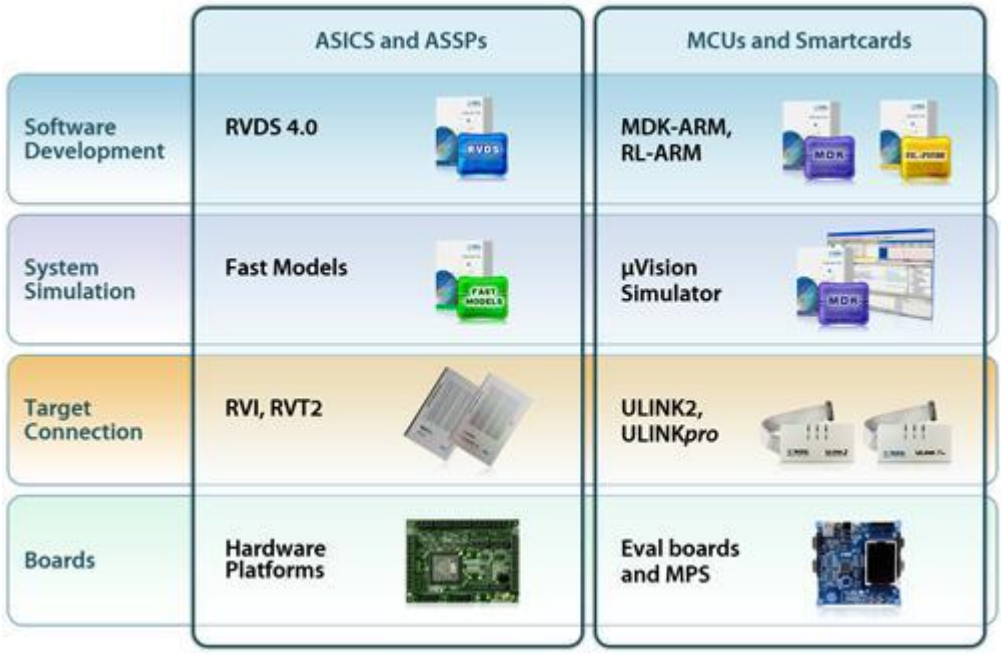
ARM, ilk başlarda işlemci olarak üretim yapmaya başlasa da sonraları işlemci üretmeyi durdurarak mimarisini başka üretici firmalara lisansını satarak günümüze kadar devam etmiştir. Bu nedenle günümüzde ARM işlemci üreten birçok firma mevcuttur.

ARM işlemcilerin Intel işlemcilere ve diğer işlemcilere göre bazı avantajları vardır;

- RISC Mimari: RISC mimarisi işlemci tasarlamayı, üretmeyi, işlemcide çalışacak kodu geliştirmeyi çok kolaylaştırıyor,
- Güç Tüketimi: ARM işlemciler donanımsal olarak diğer işlemcilere nazaran daha az transistor kullanılarak geliştirilmektedir, bu nedenle daha az güç tüketimi vardır,
- RAM'e Erişim: ARM komut seti, diğer işlemcilere kıyasla daha basit olacak şekilde tasarlanmıştır,
- Lisans Modeli-Model Çeşitliliği: İsteyen her hangi bir şirket, ARM lisansı alıp kendi ARM işlemcilerini üretebilir. Bundan dolayıdır ki piyasada çok farklı özelliklerde, fiyatlarda, farklı firmalar tarafından üretilen ARM işlemciler bulunmaktadır. Örnek olarak Apple iPad için A4 çipini kendisi geliştirmiştir.

3.2.2. ARM programlama

ARM işlemciler için yazılım geliştirme araçları, yazılımcıların ARM işlemcilerinden tam olarak faydalanabilmelerini sağlamaktadır. ARM işlemcileri en iyi performans ile kullanabilmek için bu araçları kullanmak gerekmektedir. Şekil 3.5'te bazı ARM geliştirme araçları görülmektedir.

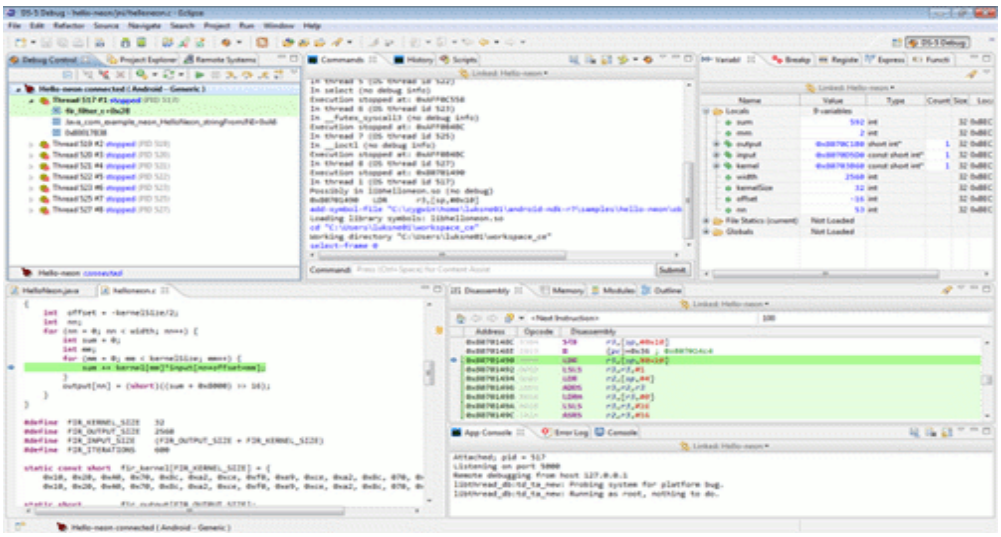


Şekil 3.5. ARM geliştirme araçları (Özkan, 2007)

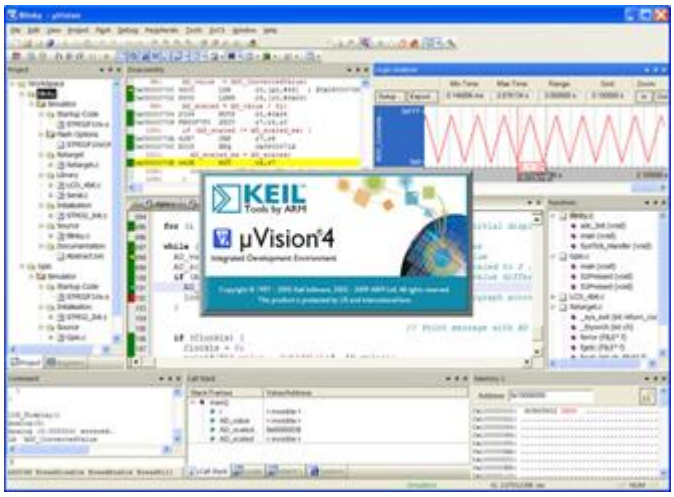
ARM yazılım geliştirme araçları 20 yıldan daha uzun bir zamandır sürekli gelişmeye devam etmektedir. ARM firması 2005 yılında şu an en iyi ARM işlemci yazılım geliştirme aracı olan Keil'i geliştirmiş ve hala da geliştirmeye devam etmektedir.

Yazılım Araçları

ARM işlemcilerde, yazılım geliştirmek için sunulan en iyi araçlar ARM Development Studio (Şekil 3.6) ve Keil (Şekil 3.7)'dir.

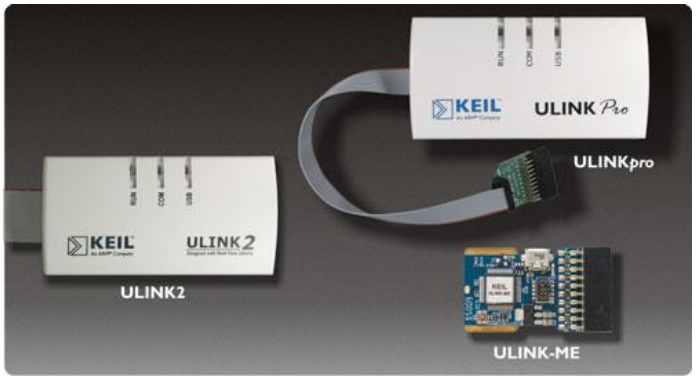


Şekil 3.6. ARM Development Studio 5



Şekil 3.7. Keil MDK-ARM™

Yazılım geliştirme aşamalarında en önemli araçlardan biride gerçek zamanlı debug yapılmasına imkân sağlayan debugger araçlarıdır. Keil firması, yazılım geliştirme aracının yanı sıra debugger üretimi de gerçekleştirmekte ve Keil yazılımı ile iyi bir şekilde çalışabilmektedir. Şekil 3.8'de Keil firmasının ürettiği ARM Debugger Adaptor görülmektedir.

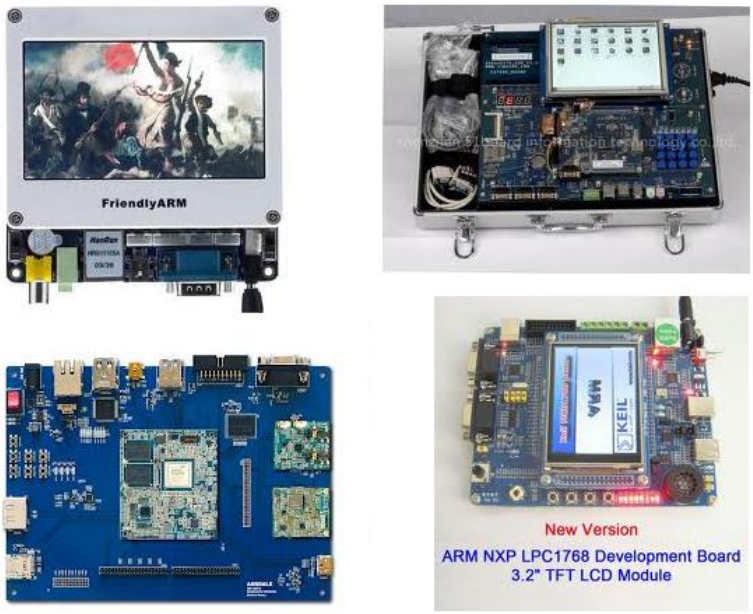


Şekil 3.8. ARM debugger adapter

ARM işlemciler için hata ayıklayıcı araçlarından bazıları şunlardır; DSTREAM, RVI, RVT2, VSTREAM, Keil ULINK araçlarıdır.

ARM Geliştirme Kartları

ARM işlemciler için birçok firmanın kendi işlemcisini ürettiği yukarıda söylenmişti. Aynı işlemcilerde olduğu gibi geliştirme kartları da birçok firma tarafından farklı özelliklerde olacak şekilde geliştirme kartları geliştirilmektedir (Şekil 3.9). Geliştirme kartları, yeni uygulamalar geliştirmeyi hızlandırmakta ve tasarımdan kaynaklı problemleri ortadan kaldırmaktadır.



Şekil 3.9. ARM geliştirme kartları

Bu tez çalışmasında Mini6410 Geliştirme kartı kullanılmıştır. Mini6410 Geliştirme kartı ARM11 mimarisine sahip SAMSUNG firmasının ürettiği S3C6410 mikrodenetleyicisine sahiptir. Sonraki bölümlerde mini6410 geliştirme kartı ayrıntılı olarak ele alınacaktır.

3.2.3. ARM11 mimarisi hakkında genel bilgiler

ARM11 Klasik ARM işlemciler kategorisinin en son geliştirilen ürünüdür. ARM11; PDA'lar, kameralar ve üst düzey cep telefonları için tasarlanmıştır. Örnek olarak, Apple iPhone'lar da ARM11 işlemci kullanılmıştır.

ARM11 işlemciler, ARMv6 komut seti ile çalışmaktadır. ARMv6 komut seti düşük güç tüketimi ile en iyi performansı sağlamaktadır.

3.2.4. Güvenlik uygulamaları

ARM11 işlemcisinde güvenli yazılım ortamı sağlaması için TrustZone(TM) yapısı geliştirilmiştir. Bu yapı işlemciyi donanımsal hasarlara karşı koruyamaz, sadece yazılımsal olarak bir güvenlik sağlayabilir. TrustZone işlemci boot ettiği anda çalışmaya başlar ve işlemci çalışmaya devam ettiği sürece tüm aşamalarda çalışmaya devam eder. TrustZone yapısı güvenli ve güvensiz durumlar olarak iki sanal işlemciden oluşur. Güvenli mod ile güvensiz mod arasında geçişler yapılabilmesi için işlemci mimarisine SMC komutu eklenmiştir. Böylece kernel ile üzerinde çalışan işletim sistemi tamamen birbirinden soyutlanmış olur. Tüm yetkileri bulunan bir saldırı programı bile kernele erişemez (Özkan, 2007).

3.2.5. Performans

ARM11, 350Mhz ile 533Mhz arasında çalışma frekansına sahiptir. ARM11 işlemcisi aynı zamanda üzerinde DSP birimi bulundurmaktadır. ARM DSP, aynı işlemci gibi kendi üzerinde register, UART, SPI gibi modüllerin bulunduğu ve 16 ya da 32 bit çekirdeğe sahip bir işlemcidir. Kullanım alanları daha çok az güç gerektiren ve batarya ile çalışan mobil platformlardır. Çünkü normal işlemcilerin 100'e yakın saat çevriminde işlem yapabildiği noktalı sayılarla olan işlemleri DSP tek saat çevriminde yapabilmektedir. Böyle olunca daha az enerji harcayacak ve pil ömürleri çok daha uzayacaktır.

3.2.6. Tek komutla çoklu veri (SIMD)

SIMD (single instruction multiple data) özellikle çoklu ortam işlemlerinde kullanılması tercih edilmektedir. ARM11'de "Advanced SIMD" kullanılmıştır. "Advanced SIMD" literatürde NEON teknolojisi olarak bilinir. 64 ve 128 bitli komut kümesi içeren işlemcilere sinyal işleme ve ses/görüntü işleme işlemlerinde hızlanma sağlamaktadır. Kapsamlı bir komut seti ve ayrı bir komut yürütme donanımı sağlamaktadır (Özkan, 2007).

MP3 kod çözme işlemini 10MHz'de ve ses kodlama kod çözme işlemini en fazla 13MHz'de yapabilir. NEON aynı anda 16 işlem yapabilir (Özkan, 2007).

SIMD'ın getirdiği bazı avantajlar bulunmaktadır. Örneğin birden çok veri ile aynı sayıyı toplanacağı varsayılsın. Bu veriler ekrandaki piksellerin her birinin tuttuğu R-G-B değerleri olabilir. Ekran parlaklığını değiştirmek için bütün piksellerin ilgili R-G-B değerlerini aynı değerle toplamak gerekecektir. Bu değerlerin hepsi için komut kullanmak yerine tek komutla bütün verileri işlemek SIMD'nin mümkün kıldığı bir çalışma biçimidir.

SIMD veriyi tek başına değil bir küme olarak algılar ve işlemek için tek veri değil birden çok veri alır. Tek komutla çalışmak döngü içerisinde çalışmaktan da iyi ve hızlı olacaktır. Bu şekilde çalışmak sonuç olarak bir hızlanma sağlamaktadır. SIMD işlemciler sadece birçok veriyi aynı anda işleyen komutları içerir.

3.2.7. Thumb® mimarisi

Thumb mimarisi 16 bitlik ARM işlemcilerin bir alt komut kümesi olarak düşünülebilir. ARM işlemcilerde bulunan 32 bitlik komutların hepsinin Thumb mimarisinde de karşılıkları vardır. Bu mimari daha az maliyet gerektiren uygulamalarda tercih edilir. 32 bit ARM komut setlerinin sunduğu tüm avantajlara sahiptir (Özkan, 2007).

Thumb 32 bit ARM mimarisine bir uzantıdır. Thumb komut seti en çok kullanılan 32 bit komutların 16 bitlik işlem kodlarına sıkıştırılmasıyla oluşturulmuş bir alt kümedir.

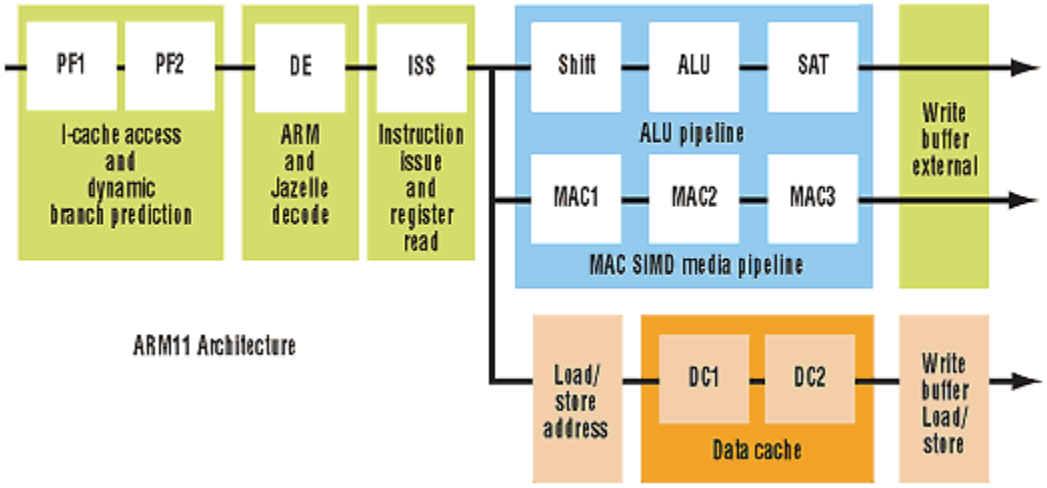
Yürütmede bu işlem kodları maliyet kaybı olmadan 32 bite geri çözülür. Böylece bir programın harcadığı bellek alanı dolayısıyla maliyet azalır. Tasarımcılar Thumb veya 32 bit ARM arasından istedikleri komut setini kullanabilir. Böylece uygulamanın gerektirdiği performans veya kod alanı kısıtlamalarında esneklik sağlanmış olunur.

"Thumb-aware" çekirdeği Thumb komutlarını çözen bir komut iş hattına sahip standart bir ARM işlemcisidir. Tasarımcı 32 bit ARM mimarisinin ve Thumb komut setinin gücünü 8 bitlik sistem maliyetiyle elde eder.

Thumb, genel 8 ve 16-bit CISC / RISC mimarilerden daha çok kod yoğunluğuna sahiptir. Thumb mimarisi Windows yazılım geliştirme ortamları tarafından desteklenmektedir.

3.2.8. İş hattı

ARM11, sekiz katlı iş hattına sahiptir. Kat sayısı arttıkça etkileşim problemleri ortaya çıkmaktadır. Ancak "forwarding" ((iş hattının sonuna gelmeden elde edilen sonucu, yazma işlemi yapmadan sonraki komuta iletebilme) ve dallanma tahminleri ile gecikmeler en aza indirgenmiştir. Şekil 3.11'de ARM11 için iş hattı yapısı görülmektedir.



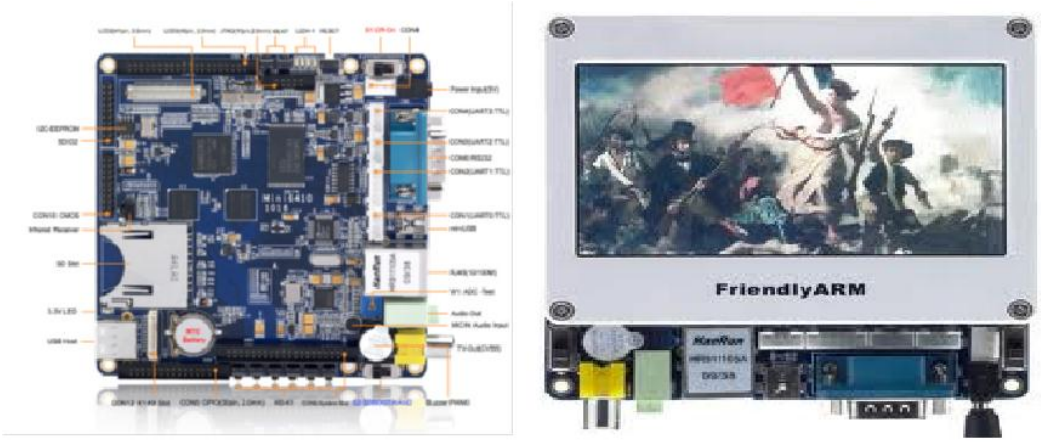
Şekil 3.11. ARM11 iş hattı yapısı (Özkan, 2007)

ARM11, Harvard mimarisi ile tasarlanmıştır. Yani veri ve komut register'ları ayrıdır. Böylece tek saat çevriminde birbirine engel olmadan hem okuma hem de yazma işlemini yapabilir.

3.3. Mini6410 Geliştirme Kartı

Mini6410 geliştirme kartı yüksek performanslı mikrodenetleyici kartıdır. Bu kart Samsung firmasının ürettiği 256 MB DDR SDRAM, 1 GB NAND flash, dâhili RTC(Real Time Clock) özelliklerine sahip S3C6410 mikro denetleyicisi kullanılarak tasarlanmıştır. Geliştirme kartı üzerinde seri port haberleşmesi için RS232 ara yüzü, USB, Ethernet, Ses giriş-çıkış, Klavye girişi, LCD dokunmatik ekran, SD kart okuyucu ve TV çıkışı arabirimlerine sahiptir. Bağlı donanımları kullanıcı isteğine göre daha da genişletilme özelliklerine sahiptir.

Bu kart Linux, Android ve Windows CE işletim sistemlerini desteklemektedir. Temel seviye donanım sürücülerini bu işletim sistemleri içermektedir. Şekil 3.12'de mini6410 geliştirme kartı gösterilmiştir.



Şekil 3.12. Mini6410 geliştirme kartı

3.3.1. Özellikleri

Samsung S3C6410 mikrodenetleyicisi; ucuz, düşük güç tüketimi, mobil uygulamalar için geliştirilmiş yüksek performanslı 32 bit RISC mimarisine sahip bir mikrodenetleyicidir. Video işleme, ses işleme için birçok donanım hızlandırıcı özelliklere sahiptir. Mini6410 geliştirme kartının tüm özelliklerine Ek F'den bakılabilir.

Mini6410 geliştirme kartına gömülü Linux işletim sisteminin ayrıntılı olarak anlatımı Ek G'de verilmiştir.

3.3.2. Otomatik giriş

Gömülü Linux sistemlerinde klavye, tuş takımı gibi donanımlar çoğunlukla bulunmaz. Bu nedenle sistem başlatıldığında işletim sistemi açılacak ve kullanıcı adı parola isteme ekranında takılı kalacaktır. Kullanıcı, harici olarak klavye ya da tuş takımı bağlamak ve parola girmek zorunda kalacaktır. Bu tip sorunların önüne geçmek için otomatik giriş ayarları yapılarak işletim sistemi açıldığında kullanıcıdan herhangi bir şey istemeden oturum başlatılmalıdır. Linux işletim sistemlerinde her türlü ayarlar dosyalarda tutulmaktadır. Otomatik giriş ile ilgili bilgiler /etc/inittab dosyasında tutulmaktadır. Bu dosyada yapılacak değişiklikler ile sisteme otomatik giriş yaptırılabilir. Ancak /etc/inittab dosyası bir sistem dosyası olduğu için işlem yapmadan önce gerekli tüm izinler verilmelidir. Daha sonra izinleri güvenlik nedeniyle eski haline getirilmesi unutulmamalıdır. Terminal penceresini açarak öncelikle root kullanıcısı için parola sormayı pasif yapmamız gerekmektedir. Bu işlem için "passwd --delete root" komutu ile parola sorma pasif hale getirilir. Daha sonra otomatik giriş için gerekli olan "mingetty" programının kurulması gerekmektedir. "sudo apt-get install mingetty" komutu ile program kurulur. Daha sonra "/etc/inittab" dosyasında "1:2345:respawn:/sbin/getty 38400 tty1", "1:2345:respawn:/sbin/mingetty --autologin root --noclear tty1" olacak şekilde değiştirilir. "shutdown -r now" komutu ile sistem yeniden başlatılır ve otomatik giriş işlemi tamamlanmış olur. Böylece mini6410 açılışta kullanıcı adı ve parola sormadan direk terminal ekranına giriş yapacaktır.

3.3.3. Açılışta kullanıcı programının doğrudan çalışması

Gömülü sistemlerde geliştirilen uygulamaların sistem boot ettikten sonra otomatik olarak çalışması gerekmektedir. Aksi takdirde kullanıcılar sistem terminali ile karşılaşacak ve sistemde olası hatalara sebebiyet verecek işlemler yapabileceklerdir. Bu gibi nedenlerden dolayı kullanıcı programları açılışta otomatik olarak başlamalı ve kullanıcı programdan başka hiçbir şey görmemelidir. Gömülü Linux sistemlerde açılış bilgileri "/etc/init.d/rc S" dosyasında tutulmaktadır. Açılışta yapılması gereken

tüm işlemler buraya yazılmalıdır. Aynı şekilde "echo metin" komutu ile açılışta ekrana verilmesi gereken mesajlar varsa yazılabilir. Fakat burada önemli olan yazılan mesajların çok fazla olmamasıdır. Çünkü uzun metinlerden oluşan mesajları FTP protokolü desteklememekte ve FTP kurulumu yapılırken bağlantı başarısız olmaktadır. Bu nedenle uzun mesajlar verilecekse bile yazılım geliştirme süreci bittikten FTP ile olan işlemler sona erdikten sonra yazılmalıdır.

Çalışması istenilen programın yolu, bu dosyanın sonuna Şekil 3.13'te görüldüğü gibi yazılarak açılışta otomatik olarak başlatılabilir.

```
#! /bin/sh
#
# rcS
#
# Call all S??* scripts in /etc/rcS.d/ in numerical/alphabetical order
#

exec /etc/init.d/rc S

/home/plg/uygulama -qws
```

Şekil 3.13. Otomatik program çalıştırma

Qt ile geliştirdiğimiz uygulamaları çalıştırmak için terminal ekranında "./dosya_yolu -qws" komutu verilmelidir. Bu komuttan önce çalıştırılacak olan dosyaya tam izinlerin verildiğinden emin olunmalıdır. "-qws" parametresi çalıştırılacak programın bir Qt uygulaması olduğunu sisteme bildirmek için kullanılır. Bu parametre olmadan çalıştırma işlemi hata ile sonlanacaktır.

3.4. Qt Derleyicisi

Qt, çapraz platformlar için C++ kodu geliştirme aracıdır. Görsel bir ara yüzü olmasından dolayı kullanımı pratiktir ve çoğunlukla tercih edilmektedir (Deepthi ve Sankaraiah, 2011). Qt, içerisinde tanımlanan zincir araçları vasıtasıyla istenilen işlemci mimarisine göre kod derleme işlemi basit bir şekilde yapabilmektedir. Bu tez çalışmasında öncelikle masaüstü bilgisayarda kodlar çalıştırılıp denenmekte, daha sonra Qt aracılığıyla çapraz olarak "ARM-Linuc-GCC toolchain" aracı ile ARM mimarisine göre kod derlenip mini6410 geliştirme kartına yüklenerek test edilmektedir.

3.4.1. Qt kurulumu

Qt geliştirme ortamı, birçok arayüz ile tümleşik olarak çalışabilir. Ancak çoğunlukla kendi tümleşik ortamı olan Qt Creator tercih edilmektedir. Qt'nin kullanılabileceği geliştirme ortamlarından bazıları şunlardır;

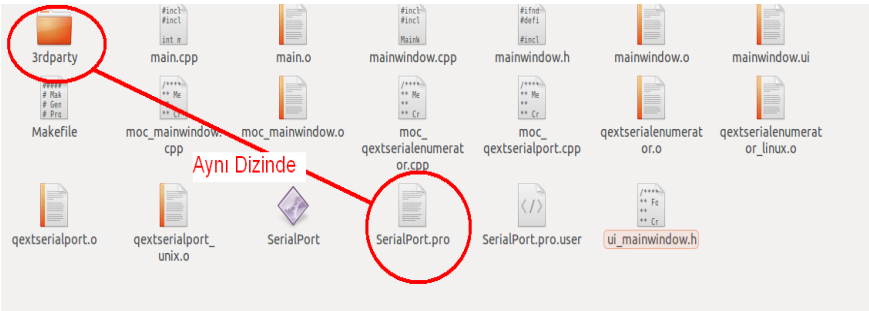
- Qt Creator
- Visual Studio
- Monkey Studio
- QDevelop
- HaiQ
- NetBeans
- Eclipse
- Code::Blocks vb…

Qt kütüphanelerinin Ubuntu işletim sistemi üzerinden yüklenmesi, proje oluşturulması, proje dosyasının düzenlenmesi, projeye harici kütüphane dosyalarının nasıl eklendiği ve çapraz derleme araçlarının programa nasıl tanıtıldığı Ek H'de ayrıntılı olarak ele alınmıştır.

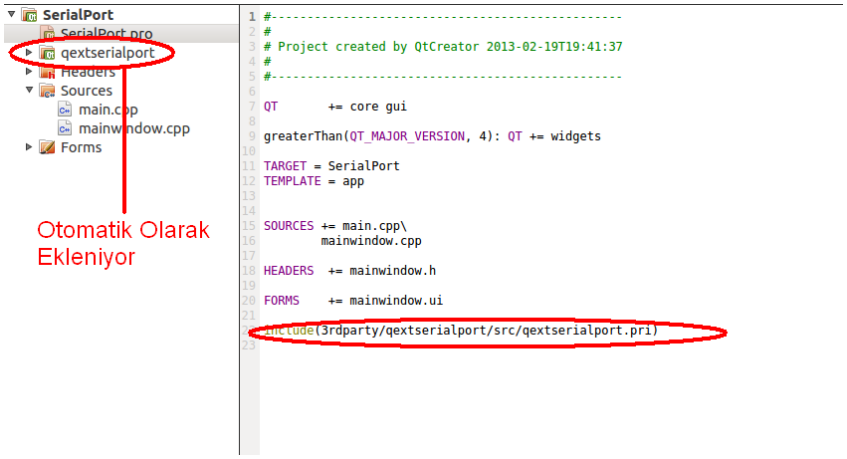
3.4.2. Qt seri port haberleşme

Qt yazılım geliştirme ortamında C++ programlama dili ile yazılmış açık kaynak kodlu birçok seri haberleşme kütüphanesi mevcuttur. Bu tez çalışmasında ***"qextserialport"*** açık kaynak kütüphanesi kullanılmıştır. Bu kütüphane kullanılmadan önce proje dosyasına eklenmelidir. Bu aşamalar sırasıyla aşağıda anlatılmıştır.

Öncelikle kütüphane dosyasını sıkıştırılmış formattan Qt proje dosyası (.pro) ile aynı dizine çıkartılır (Şekil 3.14). Daha sonra proje dosyasına ***include(3rdparty/qextserialport/src/qextserialport.pri)*** satırı eklenir. Bu işlemden sonra gerekli dosyalar projeye Şekil 3.15'te görüldüğü gibi eklenecektir.



Şekil 3.14. Seri port kütüphanesinin dizine kopyalanması



Şekil 3.15. Proje dosyasına “qextserialport” kaynak kodunun eklenmesi

Kullanılacak kütüphane, başlık dosyasına ekledikten sonra seri port ile ilgili işlemler yapılabilir. Bu işlem için kütüphane isimlerinin tanımlandığı başlık dosyasına (.h) #include "qextserialport.h" satırı eklenerek kütüphane kullanıma hazır hale getirilmelidir. Burada dikkat edilmesi gereken önemli nokta, kod derlenip çalıştırıldığında hiçbir hata vermeden seri port haberleşmesinin çalışmadığı gözlemlenebilir. Bunun sebebi derlenilen kodun, sistem yöneticisi olarak çalıştırılmamasıdır. Linux tabanlı işletim sistemleri donanıma erişmek ve kullanmak için sistem yöneticisi izinleri gerektirir. Bu nedenle derlenilen kod terminale girilerek “sudo” komutu ile çalıştırılmalıdır. Aksi durumda program çalışacak ve hiçbir hata vermeksizin seri port haberleşmesi yapmayacaktır. Bölüm 3.1.7’de anlatıldığı gibi kod “sudo” ön eki ile çalıştırılmak yerine terminale “su” komutu ile giriş yapılıp “sudo” ön eki olmadan tüm işlemler sistem yöneticisi izinleri ile çalıştırılabilir.

3.5. Görüntü İşleme

Bu tez çalışmasında, gömülü sistem için C++ dili kullanılmıştır. C++ dili ile uyumlu olarak çalışan OpenCV kütüphanesi, görüntü işleme için tercih edilmiştir.

3.5.1. OpenCV görüntü işleme kütüphanesi

OpenCV, C dilinde yazılmış, içinde C++ diline ait az sayıda sınıflar içeren gerçek zamanlı bir bilgisayarla görüntü kütüphanesidir. OpenCV ister ticari ister kişisel olsun tüm kullanımlar için ücretsiz olarak dağıtılmaktadır. OpenCV açık kaynaklı bir kütüphane olduğu için istenilen platformlar için derlenebilmektedir. Windows, Linux, Unix, MAC OS sistemler için OpenCV kütüphanesi kullanılabilir. Ayrıca OpenCV açık kaynaklı olduğu için gömülü sistemler içinde derlenip kullanılmaya uygun bir görüntü işleme kütüphanesidir.

OpenCV kütüphanelerini gömülü sistemlerde kullanmadan önce, gömülü sistemimizde bulunan işlemci mimarisine göre derlenmesi gerekmektedir. Derleme işleminden sonra OpenCV kütüphane dosyaları oluşacak ve uygulamalar kullanıma hazır hale gelecektir. OpenCV gömülü sistemler için kurulumu ayrıntılı olarak Ek I'da anlatılmıştır.

3.5.2. OpenCV temel işlemler

OpenCV kütüphanesi içinde her fonksiyonun birkaç farklı yöntemi bulunmaktadır. Bu yöntem kullanılan programlama diline göre değişiklik göstermektedir. Bu tez çalışmasında C++ programlama dili ile uygulama geliştirilmiştir. OpenCV kütüphanesinde C++ ve C dili için olan fonksiyonlar, tanımlamalar her iki dil içinde ortak bir şekilde kullanılabilir. C dili için geliştirilmiş fonksiyonlar bellek yönetimi, çöp toplama (garbage collector) gibi işlemleri de yapmaktadır. C++ dilinde bu işlemler C++ dili ile zaten yapıldığı için C++ için yazılmış fonksiyonlarda bu işlemler yapılmaktadır. Ek olarak, C dili için olan her fonksiyon ve tanımlama, C++ dili ile de çalışabilmektedir.

OpenCV kütüphanesi görüntü işlemede kullanılan tüm işlemleri yapabilecek fonksiyonlar ile donatılmıştır. Bu bölümde tez çalışmasında kullanılan görüntü işleme kütüphaneleri temel seviyede anlatılacaktır. Görüntü işlemede temel olarak bir resim okunur ve resim üzerinde işlemler yapılır. Okunacak resim bilgisayar hafızasında olan bir resim olabileceği gibi bilgisayara bağlı herhangi bir kameradan, gerçek zamanlı görüntüler alınıp, alınan görüntüler üzerinden de işlemler yapılabilir. Bu bölümde görüntü işlemede kullanılan temel işlemler (Resim okuma, kameradan görüntü alma, renk dönüşümleri, kenar belirleme, morfolojik işlemler, histogram çıkarma ve eşitleme vb.), C++ programlama dili için geliştirilmiş fonksiyonlar kullanılarak kısaca anlatılacaktır.

OpenCV kütüphanesi, uygulama geliştirilen bilgisayarın tanıdığı her kamera ile çalışabilmektedir. OpenCV kütüphanesi, kamera tanıtımı ile ilgili herhangi bir işlem yapmaya gerek kalmadan doğrudan bilgisayara bağlı bulunan kameradan görüntü alınabilir. Şekil 3.16'da görülen kod bilgisayara bağlı kameradan bir adet görüntü alarak ekranda göstermektedir.

```
#include <opencv2/highgui/highgui.hpp>
#include <opencv2/imgproc/imgproc.hpp>

using namespace cv;
using namespace std;

int main()
{
    VideoCapture cam(0);
    Mat resim;

    cam.read(resim);
    imshow("Kamera", resim);
    imwrite("sonuc.jpg", resim);
    cvWaitKey(0);
    return 0;
}
```

Şekil 3.16. Kameradan görüntü alma ve kaydetme

"VideoCapture" sınıfı parametre olarak bilgisayar bağlı kamera "ID" sini ya da okunacak video dosyasının adını almaktadır. Eğer birden fazla kamera bilgisayara bağlı ise, kullanılacak olan kameranın "ID"'si parametre olarak verilmelidir. Ayrıca bu sınıfa parametre olarak "-1" verilirse bilgisayara bağlı herhangi bir kameradan

görüntü almaktadır. Kameradan alınan görüntü istenildiği takdirde "imwrite" fonksiyonu ile bilgisayara kaydedilebilir. "imwrite" fonksiyonunda kayıt yeri belirtilmezse, görüntü, verildiği isimle, kaynak kodun bulunduğu aynı dizine kayıt edilir. "VideoCapture" sınıfının "set" fonksiyonu ile kamera özellikleri ayarlanabilmekte ve "get" fonksiyonu ile de özellik değerleri okunabilmektedir.

OpenCV'de renk dönüşümleri "cvtColor" fonksiyonu ile yapılmaktadır. OpenCV tüm renk dönüşümlerini desteklemektedir. Şekilde 3.17'de renkli olarak kameradan alınan görüntünün gri seviye dönüşüm ekran çıktısı gösterilmiştir.

Şekil 3.17. Renkli resmin gri seviye dönüşüm işlemi (a) Renkli resim (b) Gri seviye resim

Bu tez çalışmasında RGB olarak alınan görüntüler HSV (Hue, Saturation, Value) renk uzayına dönüştürülerek işlem yapılmıştır. HSV renk uzayı sırasıyla, renk özü, doygunluk ve parlaklık olarak tanımlanmaktadır. Renk özü değeri 0^o-360^o arasında değerler almaktadır. Doygunluk ve parlaklık değeri ise 0-100 arasında değişen değerler almaktadır. OpenCV kütüphanesinde H, S, V değerlerinin her biri sekiz bit ile ifade edilmekte ve dolayısıyla her biri 0-255 arasında değerler almaktadır. Denklem 3.1-3.8'de RGB renk uzayından HSV renk uzayına dönüşüm işlemi adımlarla verilmiştir.

$$R' = \frac{R}{255},\ G' = \frac{G}{255},\ B' = \frac{B}{255} \tag{3.1}$$

$$C_{max} = \max(R', G', B') \tag{3.2}$$

$$C_{min} = \min(R', G', B') \tag{3.3}$$

$$\Delta = C_{max} - C_{min} \tag{3.4}$$

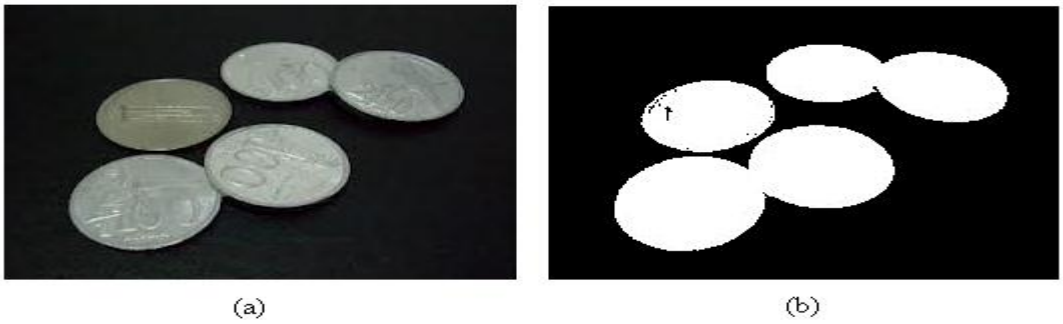
$$H = \begin{cases} 60^o \; x \left(mod6 \left(\frac{G'-B'}{\Delta} \right) \right) & , C_{max} = R' \\ 60^o \; x \left(\frac{B'-R'}{\Delta} + 2 \right) & , C_{max} = G' \\ 60^o \; x \left(\frac{R'-G'}{\Delta} + 4 \right) & , C_{max} = B' \end{cases} \tag{3.6}$$

$$S = \begin{cases} 0 & , \Delta = 0 \\ \frac{\Delta}{C_{max}} & , diğer \end{cases} \tag{3.7}$$

$$V = C_{max} \tag{3.8}$$

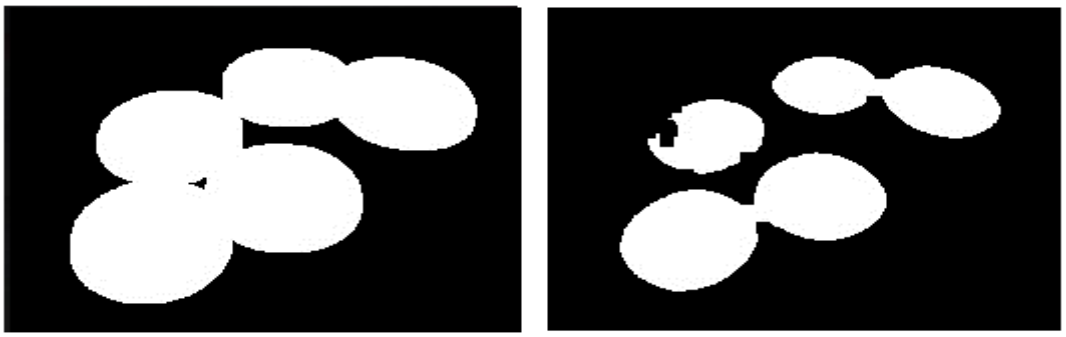
Denklem 3.1'de öncelikle RGB değerleri 0-1 arası değerler alacak şekilde $R'G'B'$değerlerine dönüştürülmektedir. Daha sonra $R'G'B'$ değerlerinin maksimum ve minimum değerleri hesaplanarak (C_{max} ve C_{min}) maksimum ve minimum arasındaki fark değeri bulunmaktadır (ΔΔ). Son olarak Denklem 3.6, 3.7 ve 3.8'de sırasıyla HSV değerleri hesaplanmaktadır.

OpenCV kütüphanesi ile ikili resme dönüşüm işlemi "threshold" fonksiyonu kullanılarak yapılmaktadır. "threshold" fonksiyonu girdi resmi, çıktı resmi, eşik değeri, maksimum değer ve mod seçimi olmak üzere beş parametre almaktadır. Kameradan alınan görüntünün "threshold" işlemi ile oluşan ekran çıktısı Şekil 3.18'de görülmektedir.



Şekil 3.18. Renkli resmin ikili resme dönüşüm işlemi (a) Renkli resim (b) 125 Eşik değeri ile ikili resim

Morfolojik işlemler, görüntü işlemede sıklıkla kullanılmaktadır. Daraltma ve genişletme işlemi olarak iki temel yöntemden oluşmaktadır. Morfolojik işlemde, resim üzerinde bir filtre ile dolaşılarak her bir piksel çevresindeki piksellerin durumuna göre genişletilir ya da daraltılır. "getStructuringElement" fonksiyonu ile resim üzerinde gezdirilecek olan filtrenin özellikleri belirlenir, "dilate" ve "erode" fonksiyonları ile de daraltma ve genişletme işlemleri yapılır. Filtre değeri varsayılan değerinde bırakılırsa, yani "erode" ve "dilate" fonksiyonuna "Mat()" parametresi gönderilirse, filtre değeri 3x3'lük bir kare matris şeklinde tanımlanır ve resme uygulanır. Daraltma ve genişletme işlemlerinin ekran çıktılarına Şekil 3.19'dan bakılabilir.

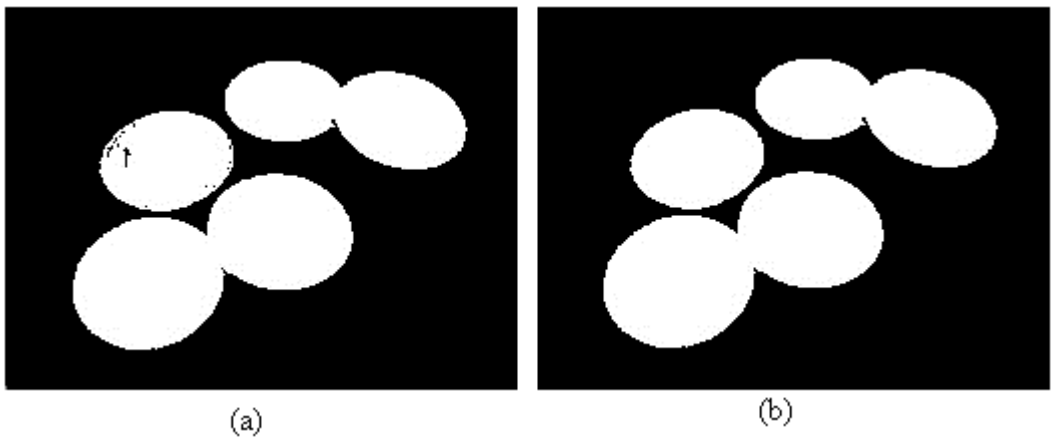


Şekil 3.19. İkili resmin genişletme ve daraltma işlemi (a) Genişletilmiş resim (b) Daraltılmış resim

Görüntü işlemede en çok kullanılan işlemlerden biri de kenar belirleme işlemleridir. Kenar belirleme işleminde; Canny, Laplacian ve Sobel v.b. yöntemler bulunmaktadır. Her yöntem kendi adı ile aynı ada sahip fonksiyonlar ile çağrılmaktadır. Canny, Laplacian ve Sobel işlemlerinin ekran çıktılarına Şekil 3.20'den bakılabilir.

Şekil 3.20. Kenar belirleme işlemi (a) Canny kenar belirleme (b) Sobel kenar belirleme (c) Laplacian kenar belirleme

Tez çalışması kapsamında "imfill" adında kapalı çevrimler içinde bulunan boşlukları doldurmak için bir fonksiyon yazılmıştır. Bu tür boşluklar morfolojik işlemlerden genişletme işlemi ile de yapılabilir ancak, kapalı çevrimin kendisi de büyüme eğilimi göstereceğinden, bu tür işlemlerde "imfill" fonksiyonu tercih edilmektedir. Şekil 3.21'de içini doldurma işlemi sonucu oluşan ekran çıktısı görülmektedir.



Şekil 3.21. İkili resmin boşluk doldurma işlemi (a) İkili resim (b) Boşluklar doldurulmuş resim

Histogram, resim içindeki piksellerin renk değerlerinin sayılarını gösteren grafiktir. Histogram eşitleme, bir resimdeki renk piksellerin belirli bir yerde kümelenmiş olmasından kaynaklanan renk bozukluklarını gidermek için kullanılan bir yöntemdir.

3.6. Robot ve Robot Kollar

Robot, temel işlemleri yerine getirebilen programlanabilen mekanik parçalardan oluşmaktadır. Robotlar; makine, elektrik-elektronik ve bilgisayar mühendisliklerinin ortak disiplin çalışmasını gerektiren cihazlardır. Robotlar programlanarak taşıma, yer yön değiştirme gibi işlevleri yerine getiren yapılardır. Robot, birbirine bağlı eklemlerden oluşan ve bu eklem noktalarına motorların bağlandığı özel amaçlar için tasarlanmış makinelerdir. Böylece, motorların hareketi ile robotların hareketi sağlanmaktadır. Robotlar hemen hemen tüm alanlarda kullanılmaktadır; tıp, endüstri ve askeri vb. alanlar.

Robot kollar insan koluna benzer şekilde tasarlanmış mekanik yapılardır. Dolayısıyla insan kolu yerine tehlike arz eden sistemlerde, çok hassas işlemlerde kullanılmaktalardır. Robot kollar kendi başlarına bir sistem olabilecekleri gibi çok daha kapsamlı bir robotun bir parçası olarak da tasarlanabilir. Robot kolların kabiliyetleri, üzerinde bulunan eksenler ile doğru orantılıdır. Ne kadar fazla eksene sahipse kullanılması ve programlanması da o kadar zordur. Eksen sayısı artıkça bir sürücü devresinin tasarlanması kaçınılmaz olur.

3.6.1. Robot kol çeşitleri

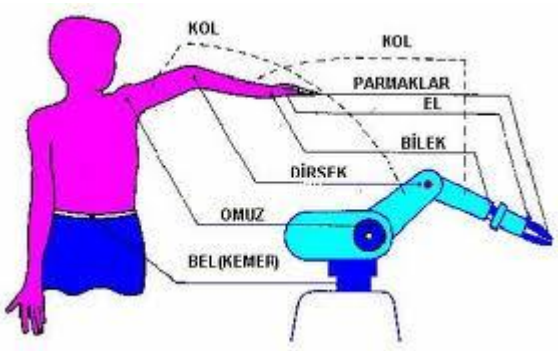
Robotlar kontrol tekniklerine, eklem yapılarına ve seri ya da paralel oluşlarına göre sınıflandırılmaktadır. Seri yapılı robotlar, eklemlerin birbirine bağlanmasıyla oluşturulan robotlardır. Geniş bir çalışma alanına sahiptirler ancak en büyük dezavantajı kendi ağırlıklarına göre çok küçük ağırlıklar üzerinde çalışabilmeleridir. Paralel yapılı robotlarda ise eklemler birden çok noktadan bağlı oldukları için kendi ağırlıklarından daha fazla ağırlıkları da taşıyabilmektedirler. Mekanik yapılarının seri yapılı robotlara göre daha karmaşık olması kontrollerinin de daha karmaşık olması anlamına gelmektedir (Bayrak, 2007). Kontrol tekniğine göre adaptif robot, adaptif olmayan robot ve akıllı robotlar olmak üzere üç gruba ayrılır. Adaptif robotlar üzerinde sensörler bulunmaktadır. Akıllı robotlar, adaptif robotlardan farklı olarak kendi bellek yapıları ve çalıştıkları çevrenin ayrıntılı bir modeli bulunmaktadır (Güzel, 2008). Eklem yapılarına göre; Kartezyen, küresel, silindirik, mafsallı eklem

yapıları kullanılmaktadır. Robotlar, mekanik, hareketlendirici ve kontrol ünitelerinden oluşmaktadır. Robotlar temel olarak beş bölümden oluşmaktadır;

- **Mekanik parçalar:** Robotun iskelet yapısının oluştuğu bölümdür.
- **Tutucu:** Robotlardaki en aktif parçadır ve en uç noktada yer alır.
- **Motorlar:** Motorları yani eklemleri ve tutucuyu hareket ettirmek için kullanılırlar. En çok servo motorlar tercih edilmektedir.
- **Kontrol Ünitesi:** Robotun yapması gereken işlemlerin programlanıp, robotu hareket ettiren birimdir.
- **Sensörler:** Robotun hareket açısı gibi durumlarda gereklidirler ancak her zaman kullanılmayabilirler.

Bu tez çalışmasında kullanılan beş eksenli ED-7220C model robot kolu mafsallı (eklemli) robot kol sınıfına girmektedir.

Mafsallı robot kollar insan koluna benzer bir yapıda tasarlanmışlardır (Şekil 3.22). Diğer robot türlerine göre daha fazla çalışma uzaylarına sahiptirler. Tüm eklemleri dönel yapıdadır. Robot kolun uç kısmına bir tutucu takılabilir (Bayrak, 2007).



Şekil 3.22. Mafsal yapılı robot kol (Bayrak, 2007).

DC Servo ve Step Motorlu Robotlar

Her iki motor tipide yüksek hassasiyet sağlar ancak düşük enerji ile daha fazla güç üretebildiği için servo motorlar daha fazla tercih edilmektedir (Bayrak, 2007). Servo motorlar hakkında ilerleyen bölümde ayrıntılı bilgi verilecektir.

3.6.2. Robot kol haberleşme sistemleri

Günümüzde endüstriyel ortamlarda kullanılan robot kolların çalışması için bir sürücü devresine ihtiyaç duyulmaktadır. Aksi halde robot kolu hareket ettirmek mümkün olmamaktadır. Bu tez çalışmasında kullanılan ED-7220C robot kolu için ED-MK4 sürücü devresi kullanılmıştır. Sürücü devreleri robot kolu hareket ettirecek bilgisayar ya da bir kumanda devresi ile farklı yollardan haberleşebilmektedir. Ethernet protokolü, paralel port, seri port gibi farklı haberleşme sistemlerini destekleyen sürücü devreler tasarlanabilmektedir. Bu çalışmada kullanılan ED-MK4 sürücü devresi seri port üzerinden haberleşmektedir.

3.6.3. ED-7220C model robot kol

ED-7220C robot kolu beş adet bağlantıdan oluşan beş eksenli bir robot koldur (Şekil 3.23). Her bir bağlantı noktası için bir adet DC servo motor kullanılmıştır. Endüstriyel sistemlerde en fazla tercih edilen robot kol beş eksenli çalışabilen robot kollardır. Ayrıca bu robot kol bir adet tutucuya sahiptir. Başlıca özellikleri şunlardır;

- Beş Eksenli çalışabilen
- Dikey olarak tasarlanmış
- +/- 0.5 mm hassasiyetlik
- Yaklaşık 100 mm/s hızda çalışabilen
- 1 kg yükleme kapasitesine sahip
- Beş adet DC servo motor
- Hareket açıları;

  Gövde Hareketi 310^{o}

  Omuz Hareketi 150^{o}

  Dirsek Hareketi 172^{o}

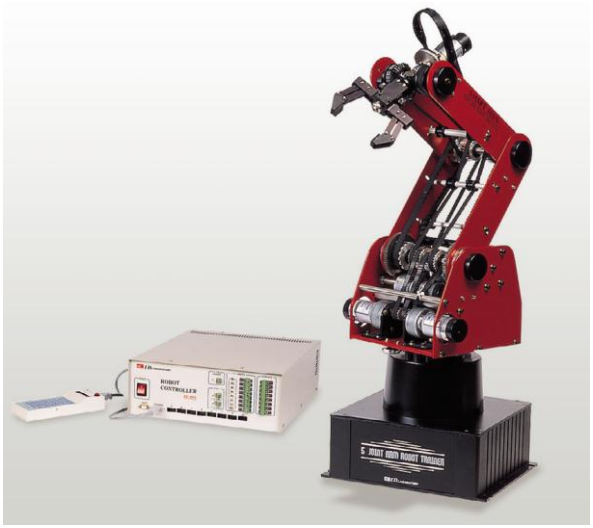
Bilek Eğilme Hareketi 260^{o}

Bilek Dönme Hareketi 360^{o}

- Tutucu açılma mesafesi 65 mm (Tutucu Koruyucusu olmadan 75 mm)
- 33 kg ağırlık

ED-MK4 Özellikleri

- 8 Port giriş/çıkış
- 16 bit işlemci
- 8 bit motor kontrol
- 8 bit uzaktan kumanda



Şekil 3.23. ED-7220C robot kol ve ED-MK4 sürücü ünitesi

3.6.3.1. Servo motorlar

Servo motorlar hız ve moment kontrolü yapan bir çeşit yardımcı motorlardır. Dakikada bir devirden az olan hareketlerinde bile kararlı çalışabilmektedirler. Servo motorlar elektronik yapılı sürücü devreler ile kontrolleri gerçekleştirilir. Hassas kullanımın ön planda olduğu sistemlerde servo motorlar tercih edilmektedir. Servo motorların tercih edilmelerinin başlıca sebepleri şunlardır;

- Servo motorların döndürme momentleri yüksektir
- Döndürme momentinin yaklaşık iki katına kadar olan değerlere yüklenebilirler ancak kısa sürelidir

- Devir sayıları kontrol edildikleri sürücü devresi ile kolay bir şekilde 1-10000 d/dk sayılarına ayarlanabilir
- Sürekli dur-kalk işlemlerine karşı dayanıklıdırlar ve sürücü devresi ile haberleşme süresi çok kısadır
- Kalkış momentleri çok güçlüdür ve ivmelenmesi diğer motor türlerine göre çok gelişmiştir.

Servo motorlar üzerinde üç adet kablo bulunmaktadır. Bunlardan biri besleme, diğeri toprak ucudur. Üçüncü kablo ise motorun kontrol edilme kablosudur. Servo motorlar kullanım alanlarına göre çeşitli büyüklükte ve güçte olabilmektedir. Hem DC hem de AC olarak çalışabilen servo motorlar mevcut olsa da en fazla DC olarak çalışan servo motorlar tercih edilmektedir.

3.6.4. ED-7220C komutları

ED-7220C sürücü devresi olan ED-MK4 devresi seri port üzerinden 9600 baud rate, 8 data bits, no parity, ve 1 stop bit ile haberleşmektedir. Haberleşme komutları Ek J'de ayrıntılı olarak gösterilmiştir.

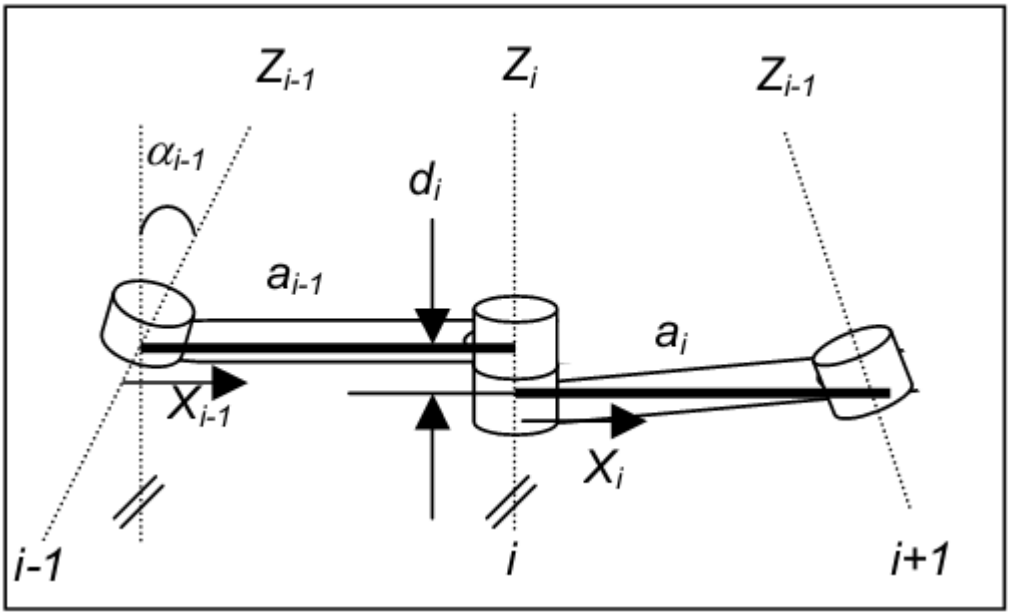
3.6.5. Robot kinematiği

Kinematik, cisimlerin, kütleleri ve cisimlere etkiyen kuvveti ele almadan sadece hareketlerin incelendiği mekanik biliminin alt dalıdır. Robot kinematiği ise robot kollarının hareketlerini inceleyen robotik bilimdir. Tasarlanan her robotun kullanılmadan önce kinematik modellerinin çıkarılması gerekmektedir. Tasarım işleminin en önemli aşamalarından biridir. Robot kinematik hesaplaması ileri ve ters olmak üzere ikiye ayrılır. İleri kinematik yöntemi, eklem uzunlukları ve eklem açıları verilen robot kolun tutucusunun üç boyutlu uzayda X, Y ve Z koordinat değerlerinin bulunması işlemidir. Ters kinematik yöntemi ise, verilen X, Y ve Z koordinat değerleri ve tutucu açısına göre her bir eklemin alacağı açı değerinin hesaplanması işlemidir. Ters kinematik yöntemi, ileri kinematik yöntemine göre daha karmaşık bir matematik hesaplaması gerektirmektedir. Ayrıca ileri kinematik yönteminde bir adet sonuç bulunmasına karşın, ters kinematik yönteminde birden fazla çözüm bulunabilir.

Robotların kartezyen koordinat sisteminde kinematik modelini çıkarmak için en yaygın olarak homojen dönüşüm yöntemi kullanılmaktadır (Küçük ve Bingül, 2004).

3.6.5.1. Homojen dönüşüm yöntemi

Robot kinematiği, en çok Denavit-Hartenberg (D-H) gösterimi ile ifade edilen homojen dönüşüm yöntemi ile ifade edilmektedir. D-H yönteminde dört ana değişken kullanılır. Bu değişkenler; iki eksen arasındaki bağ uzunluğunu ifade eden a_{i-1}, $i-1$ ile i. eksen arasındaki bağ açısını gösteren α_{i-1}, üst üste çakışan bağlar arasındaki bağ kaymasını gösteren d_i ve iki bağ arasında oluşan bağ açısını gösteren θ_i parametreleridir. Bu değişkenlere D-H değişkenleri ya da parametreleri denilmektedir (Küçük ve Bingül, 2004). D-H değişkenleri Şekil 3.24'te gösterilmiştir.



Şekil 3.24. D-H değişkenleri gösterilimi (Küçük ve Bingül, 2004)

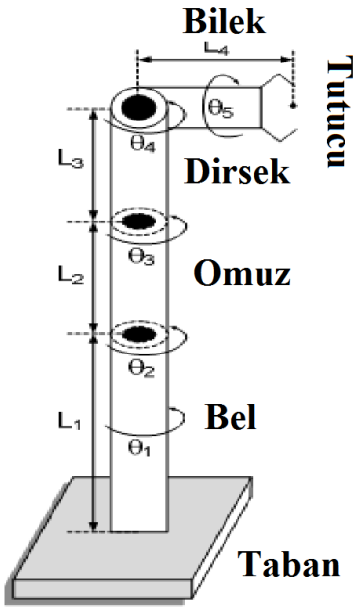
D-H gösteriminde her bir eksene üç boyutlu bir kartezyen koordinat sistemi yerleştirilir. Koordinat sisteminin Z ekseni dönme ekleminin merkezi olacak şekilde yerleştirilir. X ekseni Z eksenine dik olacak şekilde bağ uzunluğu yönündedir. Y ekseni ise sağ el kuralına göre tespit edilmektedir.

Denklem 3.9'da D-H değişkenleri ile bir eklemin homojen dönüşüm matrisi gösterilmiştir.

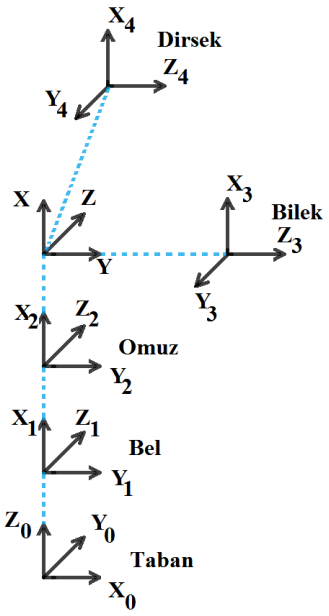
$$ {}^{i-1}_{i}T = \begin{bmatrix} cos\theta_i & -sin\theta_i & 0 & a_{i-1} \\ sin\theta_i cos\alpha_{i-1} & cos\theta_i cos\alpha_{i-1} & -sin\alpha_{i-1} & -sin\alpha_{i-1} d_i \\ sin\theta_i sin\alpha_{i-1} & cos\theta_i sin\alpha_{i-1} & cos\alpha_{i-1} & cos\alpha_{i-1} d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.9) $$

Burada i değeri hesaplanmak istenen bağlantı noktasını ifade etmektedir. a değişkeni ardışık iki eklem arasındaki ölçülen mesafe, α değişkeni ardışık iki eklem arasındaki eksen açı değeri, d değişkeni üst üste çıkan bağlar arsındaki eklem kayması değeri ve θ değişkeni ardışık iki eklem arasındaki açı değerini temsil etmektedir. Şekil 3.24'te verilen değişkenler görsel olarak takip edilebilir.

Şekil 3.25'te ED-7220C robot kol için kinematik model ve Şekil 3.26'da Kartezyen koordinat sistemi gösterilmiştir.



Şekil 3.25. ED-7220C robot kol kinematiği (Iqbal vd., 2012)



Şekil 3.26. ED-7220C kartezyen koordinat sistemi (Iqbal vd., 2012)

Çizelge 3.1'de ED-7220C robot kol için D-H değişkenleri, Çizelge 3.2'de eklem uzunlukları ve Çizelge 3.3'te de her bir eklemin dönebileceği açı değerleri gösterilmiştir.

Çizelge 3.1. ED-7220C robot kol D-H değişkenleri

i	θ_i	α_{i-1}	a_{i-1}	d_i
1	θ_1	0	0	L_1
2	θ_2-90^o	$-\pi/2$	0	0
3	θ_3	0	L_2	0
4	θ_4	0	L_3	0
5	θ_5	$-\pi/2$	0	0
6	0	0	0	L_4

Çizelge 3.2. ED-7220C robot kol eklem uzunlukları

Eklem	Bel	Omuz	Dirsek	Bilek
Sembol	L_1	L_2	L_3	L_4
Uzunluk (mm)	385	220	220	155

Çizelge 3.3. ED-7220C robot kol eklem açı değerleri

Eklem	Bel	Omuz	Dirsek	Bilek (Eğilme)	Bilek (Dönme)
Aralık	$-244^o/66^o$	$-120^o/30^o$	$-106^o/66^o$	$-220^o/40^o$	$0^o/360^o$
Toplam	310^0	150^0	172^0	260^0	360^0

İleri kinematik hesabı için her bir eklemin D-H değişkenlerine göre homojen dönüşüm matrisi hesaplanarak birbiri ile çarpılır. Denklem 3.10'da ED-7220C robot kol için ileri kinematik hesaplama formülü verilmiştir.

$$ {}^{6}_{0}T = {}^{0}_{1}T\,{}^{1}_{2}T\,{}^{2}_{3}T\,{}^{3}_{4}T\,{}^{4}_{5}T\,{}^{5}_{6}T \tag{3.10} $$

$$ X = \cos(\theta_1)[L_2\cos(\theta_2) + L_3\cos(\theta_2+\theta_3) - L_4\sin(\theta_2+\theta_3+\theta_4)] \tag{3.11} $$

$$ Y = \sin(\theta_1)[L_2\cos(\theta_2) + L_3\cos(\theta_2+\theta_3) - L_4\sin(\theta_2+\theta_3+\theta_4)] \tag{3.12} $$

$$ Z = L_1 - L_2\sin(\theta_2) - L_3\sin(\theta_2+\theta_3) - L_4\cos(\theta_2+\theta_3+\theta_4) \tag{3.13} $$

İleri kinematik hesaplama yönteminde ED-7220C beş eksenli robot kol için verilen açı değerlerine göre X, Y ve Z konum bilgileri sırasıyla Denklem 3.11, 3.12 ve 3.13'te verilmiştir.

Ters kinematik hesaplama yönteminde, nesnenin robot kolunun çalışma uzayı içinde olduğundan emin olunmalıdır. Aksi takdirde ters kinematik hesaplanamayacaktır. Ters kinematik yönteminde tutucunun açısı (θ_5), X,Y ve Z koordinat bilgileri ile verilmelidir. Ters kinematik hesaplama öncelikle θ_1 değeri hesaplanır (Denklem 3.15), θ_1 kullanılarak θ_3 değeri hesaplanır (Denklem 3.16), θ_1 ve θ_3 değerleri kullanılarak θ_2 değeri hesaplanır (Denklem 3.19) ve son olarak θ_4 değeri hesaplanır (Denklem 3.22).

$$Atan2(x,y) = \begin{cases} \arctan\left(\frac{y}{x}\right) & x > 0 \\ \arctan\left(\frac{y}{x}\right) + \pi & y \geq 0, x < 0 \\ \arctan\left(\frac{y}{x}\right) - \pi & y < 0, x < 0 \\ +\frac{\pi}{2} & y > 0, x = 0 \\ -\frac{\pi}{2} & y < 0, x = 0 \\ tanımsız & y = 0, x = 0 \end{cases} \quad (3.14)$$

$$\theta_1 = Atan2(p_x, p_y) \quad (3.15)$$

$$\cos\theta_3 = \frac{(\cos\theta_1 p_x + \sin\theta_1 p_y)^2 + (p_z - L_1)^2 - L_2^2 - L_3^2}{2L_2L_3} \quad (3.16)$$

$$\sin\theta_3 = \mp\sqrt{1 - \cos{\theta_3}^2} \quad (3.17)$$

$$\theta_3 = Atan2(\sin\theta_3, \cos\theta_3) \quad (3.18)$$

$$\cos\theta_2 = \frac{(\cos\theta_1 p_x + \sin\theta_1 p_y)(\cos\theta_3 L_3 + L_2) - (p_z - L_1)\sin\theta_3 L_3}{(\cos\theta_3 L_3 + L_2)^2 + \sin{\theta_3}^2 {L_3}^2} \quad (3.19)$$

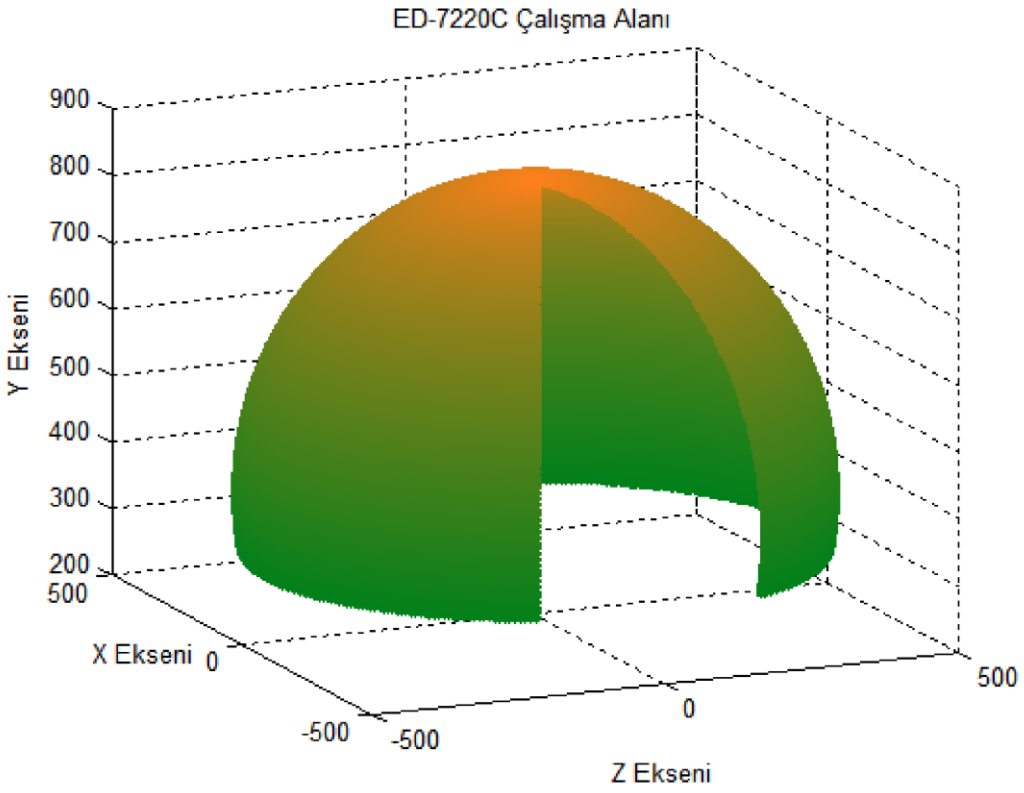
$$\sin\theta_2 = -\frac{(\cos\theta_1 p_x + \sin\theta_1 p_y)\sin\theta_3 L_3 + (p_z - L_1)(\cos\theta_3 L_3 + L_2)}{(\cos\theta_3 L_3 + L_2)^2 + \sin{\theta_3}^2 {L_3}^2} \quad (3.20)$$

$$\theta_2 = Atan2(\sin\theta_2, \cos\theta_2) \quad (3.21)$$

$$\theta_4 = \theta_5 - (90^o + \theta_2 + \theta_3) \quad (3.22)$$

Yukarıdaki denklemlerde p_x, p_y ve p_z değerleri sırasıyla üç boyutlu uzayda X, Y ve Z değerlerini ifade etmektedir.

Şekil 3.27'de ED-7220C robot kolun üç boyutlu uzayda çalışma aralığı gösterilmiştir.



Şekil 3.27. ED-7220C robot kolu üç boyutlu çalışma alanı

Şekil 3.27'den de görüldüğü gibi tabana bağlı olan eklemin hareket açısı (θ_1) 310^o olduğundan dolayı tarayamadığı alan bulunmaktadır.

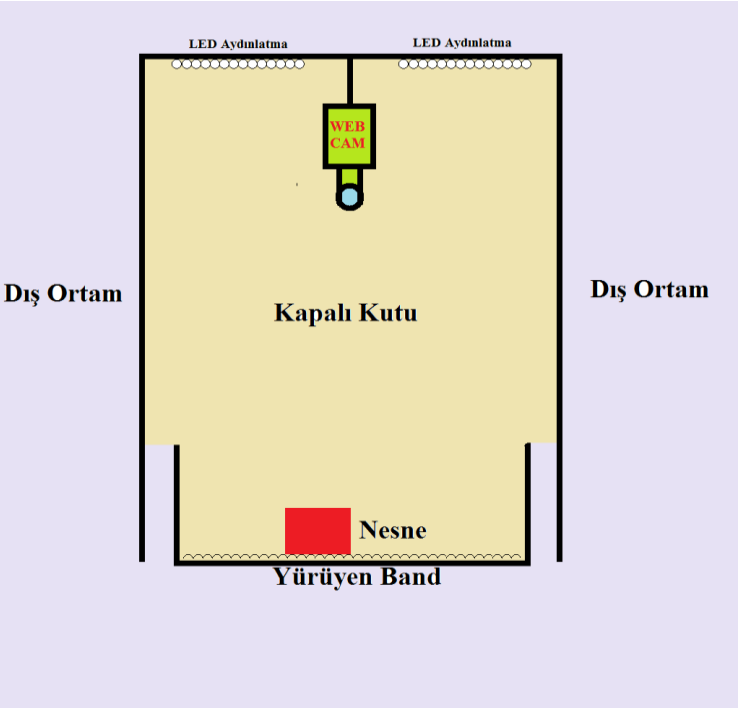
4. ARAŞTIRMA BULGULARI

Tez çalışmasında uygulanan adımlar başlıklar halinde ayrıntılı olarak anlatılmıştır.

4.1. Kameradan Görüntü Alma İşlemi

Bu tez çalışmasında A4 Tech web tabanlı kamera kullanılmıştır. Her görüntü işleme projesinde en temel işlem, sisteme bağlı bir kameradan görüntü almak ve işlemektir. Bu tez çalışmasında da gerçek zamanlı olarak sistemden görüntü alınmakta ve işlenerek robot kola yapması gerekenler bildirilmektedir. Çalışmada kullanılan resimlerin çözünürlüğü 180x240 olarak belirlenmiştir.

Görüntü işlemede en önemli olan aşamalardan biri aydınlatma işlemidir. Bu tez çalışmasında Şekil 4.1'de görüldüğü gibi, yürüyen bant üzerine kapalı bir düzenek kurulmuş, düzeneğin içi LED aydınlatma sistemi ile aydınlatılmıştır. Kamera bu düzenek içerisine yerleştirilmiş ve gerçek zamanlı olarak, dış çevrenin ışık şiddetinden bağımsız olarak sürekli olarak aynı ışık şiddetinde görüntüler alınarak mini6410 geliştirme kartına iletilmiştir.



Şekil 4.1. Kapalı kutu kamera sistemi

Ayrıca kameradan görüntü alındığı esnada, nesnelerin yürüyen bant üzerinde olabileceği alanların dışında kalan bölgeler, görüntü işleme ile kırpılarak hesaplama dışı bırakılmıştır. Böylece hesaplamalarda gereksiz bölgelerle program ilgilenmeyip hızlanma sağlanmıştır.

4.2. Renk Dönüşüm ve Eşikleme İşlemi

Geliştirilen sistemde, üretim bandı üzerinde bulunan ürünlerin rengi kırmızı olarak belirlenmiştir. Dolayısıyla ürün ayrıştırma işleminde, görüntüde bulunan kırmızı bölgeler ilgi odağı olmuştur. Bu nedenle, kameradan alınan görüntüler belli değerlere göre karşılaştırılmış ve belirlenen aralık dışında olan bölgeler elenmiştir. Çalışmada, kameradan alınan görüntüler, RGB formatında HSV formatına dönüştürülmüştür. HSV formatı RGB formatına göre daha etkili bir renk ayrıştırma performansı sağlamaktadır. HSV formatına dönüştürülen resimlerde Sırasıyla H, S, ve V değerleri, kırmızı rengin dışında bulunan bölgeleri elemek için en ideal değerler tespit edilerek resimler ikili forma dönüştürülmüştür. Bu tez çalışmasında kırmızı renkleri ayrıştırmak için kullanılan H, S, V değer aralıkları aşağıda verilmiştir.

$0 < \mathrm{H} < 255$

$55 < \mathrm{S} < 255$

$0 < \mathrm{V} < 255$

HSV dönüşüm işlemi sonunda resim ikili forma dönüştürülmektedir.

4.3. Morfolojik İşlemler ve Boşluk Doldurma

Renk uzayı dönüşüm işleminden sonra ikili forma dönüşen resimde, özellikle nesnelerin kenarlarında bulunan açıklıkları gidermek için morfolojik işlemler kullanılmıştır. Bunun için bir defa genişletme, iki defa daraltma işlemi resme uygulanarak kenarlarda olan boşluklar tamamlanmıştır.

Morfolojik işlemler sonunda nesnelerin içinde bulunan boşluklar geliştirilen kod ile doldurulmuştur. Böylece nesneler ikili formda bütün olarak elde edilmiş olmaktadır.

Kamera açısına gelen nesnelerin, görüntü içine tamamının girdiğini anlamak için, ikili forma dönüştürülen resimlerin, matris olarak en üst satırlarındaki piksel

değerleri toplanmaktadır. Toplam değer sıfır olduğu sürece herhangi bir ürünün gelmediği, toplam değerin sıfırdan büyük olduğu değerlerde ise ürünün geldiği ve kamera açısına tam olarak girmediği anlaşılmaktadır. Ürün geldiği tespit edildikten sonra tekrar matrisin en üst satırındaki piksel değerleri toplamı sıfır olduğunda ürünün kamera açısına tam olarak girdiği anlaşılmaktadır. Bu aşamadan sonra ürün tespit işlemi (ÜTİ) gerçekleştirilmektedir. Ürünler kamera açısına girdiği anda, resmin bütününde piksel değerleri toplamı (PDT) hesaplanmaktadır. Kullanılan çözünürlük değerine göre toplam değer farklılık göstermektedir. Bu çalışmada piksel değerleri toplamı 50000 ve üzeri ise hatalı ürün (HÜ), 50000'den küçük ise hatasız ürün (HSÜ) olarak tespit edilmektedir. Hatalı ürün tespit edildiğinde robot kola seri port üzerinden gerekli bilgi gönderilerek hatalı ürün bant üzerinden alınmaktadır. Denklem 4.1'de hesaplama formülü gösterilmiştir.

$$\text{ÜTİ} = \begin{cases} \text{HÜ}, & eğer\ \text{PDT} > 50000 \\ \text{HSÜ}, & diğer \end{cases} \tag{4.1}$$

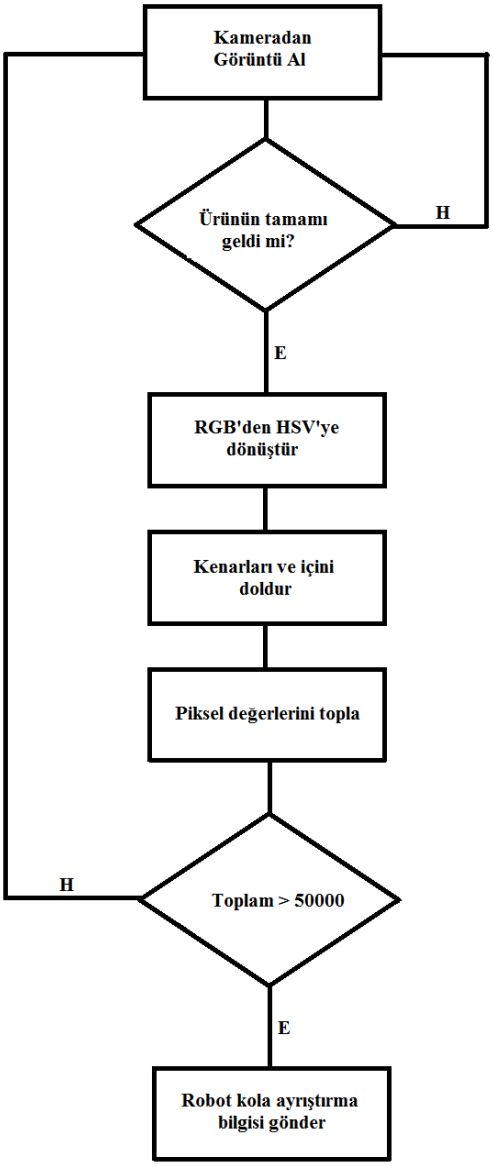
4.4. Robot Kol Parça Ayırma İşlemi

Bu tez çalışmasında robot kolun hareket edeceği üç temel konum vardır: Başlangıç konumu, parça alma noktası ve parça bırakma noktası. Uygulama geliştirildikten sonra her bir konumun koordinat değerleri hesaplanmış ve kod içinde belirtilmiştir. Sistem ilk başladığı anda robot kol başlangıç konumunda bulunmaktadır. Uygulama başlatıldığı zaman robot kol otomatik olarak hatalı ürünlerin alınacağı parça alma noktasına konumlandırılmaktadır. Tutucu açık vaziyette mini6410 geliştirme kartından gelecek bilgiyi beklemektedir. Mini6410, hatalı ürünü tespit ettikten sonra, hatalı ürünün robot kolun tutucusuna ulaşması için 18 saniye geçmesi gerekmektedir. Mini6410 robot kola bilgiyi göndermek için, her hatalı ürünü tespit ettikten sonra 18 saniye beklemekte ve robot kola bilgiyi göndermektedir.

Robot kol, kendisine bilgi gelince tutucuyu kapatmakta, ürünü tutmakta ve parça bırakma konumuna hareket edip, tutucuyu açarak tuttuğu parçayı bırakmaktadır. Bu işlemden sonra tekrar parça alma noktasına giderek, mini6410'dan gelecek bilgiyi beklemektedir. Uygulama kapatıldığı zaman robot kol otomatik olarak başlangıç konumuna hareket etmektedir. Aksi takdirde, robot kol, elektrik enerjisi verildiği

andaki konumunu başlangıç noktası olarak algılayacak ve hesaplanan konum koordinatları farklı konumları ifade edecektir. Bu nedenle uygulama kapatıldığı zaman robot kola başlangıç konumuna hareket bilgisi gönderilmesi gerekmektedir.

Yukarıda anlatılan tüm işlemler Şekil 4.2'de akış diyagramı olarak gösterilmiştir. Ayrıca uygulamaya ait resimlere Ek K'dan bakılabilir.



Şekil 4.2. Görüntü işleme ve beş eksenli robot kol ile üretim bandında nesne denetimi işlemi akış diyagramı

5. TARTIŞMA VE SONUÇLAR

Bu çalışmada, gömülü sisteme bağlı bir kameradan görüntü alarak, görüntü işleme tekniklerine tabi tutulmuş ve üretim bandı üzerinde hareket eden ürünler tasniflenmiştir. İncelenen ürünler hatalı olduğu tespit edildiğinde gömülü sisteme bağlı robot kola gerekli komutlar gönderilerek hatalı ürün üretim bandından alınarak hatalı ürün sepetine konulmuştur.

ED-7220C model robot kol saniyede 100 mm yol almaktadır. Bu şartlar dâhilinde bir ürünün üretim bandından alınıp, hatalı ürün sepetine konulması işlemi yaklaşık 18 saniye sürmektedir. Kısaca dakikada üç ürün ayrıştırılabilmektedir. Geliştirilen sistemde hata payı yoktur. Yani, yanlışlıkla, hata olmayan ürün ayrıştırılması söz konusu değildir. Ancak yukarıda bahsedildiği üzere 18 saniyede bir robot kol ürün ayrıştırma işlemine odaklanabilmektedir. Bu nedenle daha kısa sürelerde art arda gelen ürünler olduğu zaman sistem arada gelen ürünleri ayrıştıramamaktadır. Bu sorunun çözümü için aynı robot koldan üç adet belirli aralıklarla yan yana yerleştirilirse her hangi bir robot kolun kaçırdığı ürünü diğer robot kollar yakalayıp ayıklayabilecektir. Ancak sistem maliyeti hesaplandığında üç adet ED-7220C robot kullanılmasın yerine daha hızlı çalışan bir tek robot kol tercih edilmelidir.

KAYNAKLAR

Aby, P.K., Jose, A., Jose, B., Dinu, L.D., John, J., Sabarinath, G., 2011. Implementation and Optimization of Embedded Face Detection System, International Conference on Signal Processing, Communication, Computing and Network Technologies (ICSCCN), Thuckafay, s. 250-253.

Ali, H., Seng, T.C., Hoi, L.H., Elshaikh, M., 2012. Development of Vision-Based Sensor of Smart Gripper for Industrial Applications, IEEE 8th International Colloquium on Signal Processing and its Applications, Melaka, s. 300-304.

Anh, L.T., Song, J.B., 2010. Object Tracking and Visiual Servoing using Features Computed from Local Feature Descriptor, International Conference on Control, Automation and Systems, Gyeonggi-do, Korea, s. 1044-1048.

Basile, F., Caccavale, F., Chiacchio, P., Coppola, J., Curatella, C., 2012. Task-Oriented Motion Planning for Multi-Arm Robotic Systems, Robotics and Computer-Integrated Manufacturing, s. 569-582

Bayrak, A., 2007. Beş Eksenli Bir Robot Kolunun Simülasyonu ve Kontrolü, Yüksek Lisans Tezi, Gazi Üniversitesi Fen Bilimleri Enstitüsü, Ankara.

Bayrak, A., Sarıtaş, M., 2008. Beş Eksenli Bir Robot Kolu Simülasyonu ve Engel / Hedef Takibi, ELECO, Bursa, s. 1-5.

Bejo, A., Pora, W., Kunieda, H., 2009. Development of a 6-Axis Robotic Arm Controller Implemented on s Low-Cost Microcontroller, 6th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), Pattaya, Choburi, s. 328-331.

Chang, H.C., Shih, T.M., 2008. Visual Servo Control of a Three Degree of Freedom Robotic Arm System, IEEE International Conference on Advanced Robotics and its Social Impacts, Taipei, Taiwan, s. 1-6.

Deepthi, R.S., Sankaraiah, S., 2011. Implementation of Mobile Platform Using Qt and Open CV for Image Processing Applications, IEEE Conference on Open Sysyems (ICOS),Langkawi, Malaysia, s. 284-289.

Fanhua, Y., Dengfeng, W., Hongwei, Z., Zhiqiang, Z., 2010. Video Image Acquisition System Based on Embedded Linux, International Conference on Digital Manufacturing & Automation, ChangSha, s. 907-909.

Güzel, M.S., 2008. Altı Eksenli Robot Kolun Hareketsel Karakteristiğinin Görsel Programlanması ve Gerçek Zamanlı Uygulamalar, Yüksek Lisans Tezi, Ankara Üniversitesi Fen Bilimleri Enstitüsü, Ankara.

Huang, G.S., 2011. Robotic Arm Grasping and Placing Using Edge Detection System, 8th Asian Control Conference (ASCC) Kaohsiung, Taiwan, s. 960-964.

Iqbal, J., Islam, R., Khan, H., 2012. Modelin and Analysis of a 6 DOF Robotic Arm Manipulator, Canadian Journal on Electrical and Electronics Engineering, Canada, s. 300-306

Islam, R., Iqbal, J., Manzoor, S., Khalid, A., Khan, S., 2012. An autonomous image-guided robotic system simulating industrial applications, 7th IEEE International Conference on System of System Engineering (SoSE), Genova, Italy, s. 314-319.

Jiao, G.S., Qin, H.B., 2011. Design of Remote Image Monitoring System Based on Embedded Linux and OMAP3530, International Conference on Electronics, Communications and Control, Ningbo, s. 2017-2019

Küçük, S., Bingül Z., 2004. Robot Sistemlerinde Kinematik Yöntemlerin Araştırılması, Journal of Polytechnic, Ankara, s. 107-117

Liping, Y., Kai, S., 2010. Design and Realization of Image Processing System Based on Embedded Platform, International Forum on Information Technology and Applications, Kunming, s. 446-449.

Liu, J., Chou, W., 2009. Design and Implimentation of an Embedded Image Capture Device for Mobile Robots, 2nd International Congress on Image and Signal Processing (CISP), Tianjin, s. 1-5.

Low, S.M., 2010. A Wireless Webcam-based Robotic Vision System, International Conference on Intelligent and Advanced Systems (ICIAS), Kuala Lumpur, Malaysia, s. 1-4.

Luo, R.C., Tsai, C.H., Lai, C.C., Chang, C.M., 2008. Autonomous Security Robot Services Using Eye-in Hand Visual Servo Systems, IEEE International Conference on Advanced Robotic and its Social Impacts, Taipei, Taiwan, s. 1-6.

Ogawa, M., Shimizu, S., Kadogawa, T., Hashizume, T., 2011. Development of Air Hockey Robot Improving with the Human Players-Arm Control of Effective Quick Draw Taking An Opponent's Eye Movement into Account, 37th Annual Conference on IEEE Industrial Electronics Society (IECON), Melbourne, VIC, s. 3364-3369

Ozkan, E., 2007. ARM11 Mimarisi. İstanbul Teknik Üniversitesi, Dönem Projesi, 17s, İstanbul.

Seelye, M., Gupta, G.S., Seelye, J., Mukhopadhyay, S.C., 2010. Camera-in-hand Robotic System for Remote Monitoring of Plant Growth in a Laboratory, IEEE Instrumentation and Measurement Technology Conference (I2MTC), Austin, TX, s. 809-814.

Rapp, H.H., 2011. A Ping-Pong Ball Catching and Juggling Robot, a Real-Time Framework for Vision Guided Acting of an Industrial Robot Arm, Proceeding of the 5th International Conference on Automationi Robotics and Applicationas, Wellington, New Zealand, s. 430-435.

Wang, H., You, X., Wang, R., 2012. Design of Image Capture and Transmission Embedded System for Remote Monitoring, 8th International Conference on Information Science and Digital Content Technology, Jeju Island, Korea (South), s. 661-664.

Wei, Z., Cai, L., 2010. Design for Motion Detection System Based on Embedded Linux, International Conference on Multimedia Technology, Ningbo, s. 1-3.

Zhang, X., Li, X., Lu, K., 2012. Remote Video Monitoring System Based on ARM and Linux, Third International Conference on Intelligent Control and Information Processing, Dalian, China s. 369-372.

Zheng, G.R., Xiong, C.Z., Zhang, Y., 2011. A Method of Embedded Video Surveillance based on OpenCV, International Conference on E –Business and E –Government (ICEE), Shanghai, China, s. 1-4

EKLER

EK A. arm-linux-gcc-4.5.1 aracının sisteme entegre edilmesi

EK B. Mini6410 için Linux çekirdeğinin derlenmesi

EK C. Mini6410 Geliştirme kartı kök dosya sistemi oluşturma

EK D. Linux tabanlı sistemlere FTP ve SSH kurulumu

EK E. Temel Linux komutları

EK F. Mini6410 geliştirme kartı özellikleri

EK G. Mini6410 gömülü Linux kurulumu

EK H. Qt kurulumu ve ayarlarının yapılması

EK I. OpenCV gömülü sistemler için kurulumu

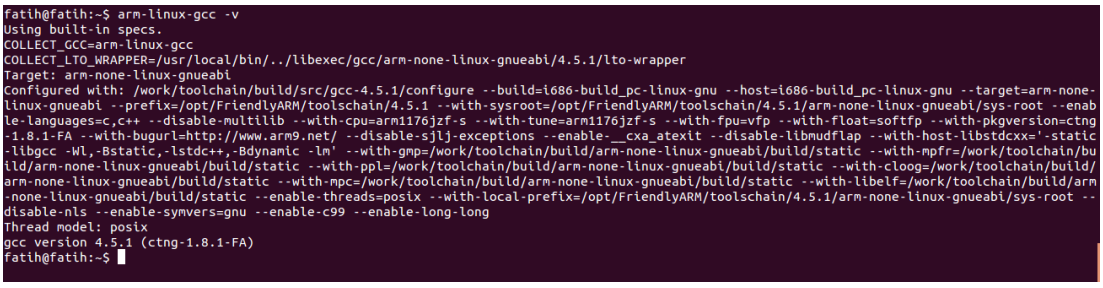
EK J. ED-MK4 Sürücü devresi seri port haberleşme komutları

EK K. Uygulama resimleri

EK A. arm-linux-gcc-4.5.1 aracının sisteme entegre edilmesi

Çapraz derleme işlemlerini yapabilmek için öncelikle uygulama geliştirilecek olan bilgisayara çapraz derleme için gerekli olan araçların kurulması gerekmektedir. Bu tez çalışmasında ARM tabanlı bir geliştirme kartı kullanıldığından dolayı ARM işlemciler için gerekli olan çapraz derleme araçları kullanılması gerekmektedir. ARM işlemciler birçok firma tarafından üretildiği için, birçok çapraz derleme aracı bulunmaktadır. Bu tezde arm-linux-gcc-4.5.1 çapraz derleme aracı kullanılmıştır. Bu araç mini6410 geliştirme kartının üretici firması internet sitesinden indirilebilir.

arm-linux-gcc-4.5.1 aracı internetten indirildikten sonra sıkıştırılmış dosya içinden "arm-none-linux-gcc" ve "bin" dizinleri masaüstü bilgisayarın /usr/local dizinine kopyalanmalıdır. Linux sistemlerin path'inde varsayılan olarak /usr/local dizini kayıtlıdır. Böylece "arm-linux-gcc toolchain"'i path'e otomatik olarak eklenmiş olacaktır. Bu dosyalar farklı bir dizine çıkarılıp, dizin yolu path değişkenine eklenerek de aynı işlem yapılabilir. Bu işlemin doğru yapıldığını test etmek için terminalde "arm-linux-gcc –v" komutu yazarak sonuç gözlenir. Şekil A.1'de görüldüğü gibi versiyon hakkında bilgiler veriliyorsa yapılan işlem doğru bir şekilde gerçekleşmiş demektir. Aksi halde aynı işlemlerin doğru bir şekilde tekrarlanması gerekmektedir.



Şekil A.1. Çapraz derleme versiyon bilgisi

EK B. Mini6410 için Linux çekirdeğinin derlenmesi

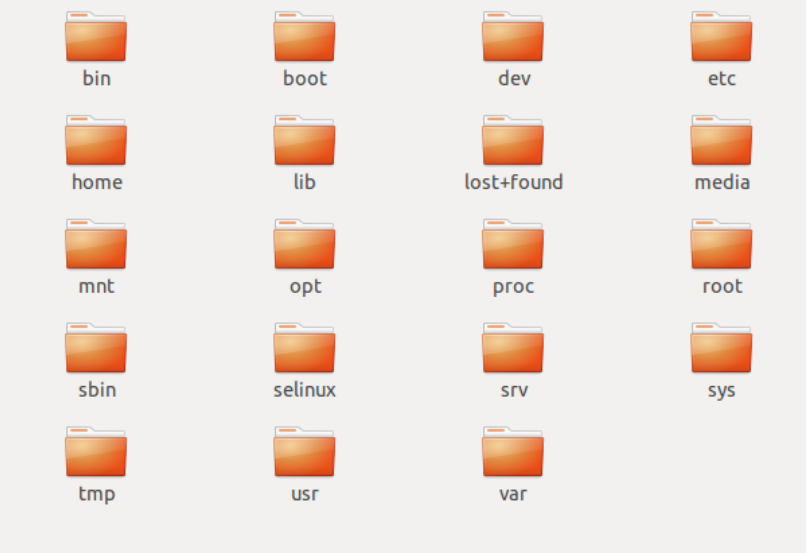
Linux çekirdeği açık kaynak olarak ücretsiz bir şekilde kullanıcılara dağıtılmaktadır. Açık kaynak olması, kullanıcıya esneklik getirmektedir. Yani derleme işlemini kullanıcı yaparak, istediği özellikleri ekleyip, istemediği özellikleri çekirdekten çıkarabilmektedir. Böylece uygulamalar için gerekli olan optimum özellikli ve boyutlu bir çekirdek derlenebilmektedir.

Uygulamada kullanılmak istenen çekirdek versiyonu belirlendikten sonra internet sitesinden indirilir ve Ubuntu sisteminde terminal penceresi açılarak çekirdeğin indirildiği dizine gelinir. Tüm işlemler için /home dizini altında "mini6410" adında ortak bir dizin oluşturmak ve tüm işlemleri bu dizin üzerinden gerçekleştirmek için iyi bir yöntemdir. Böylece dizin ve dosyalar düzenli bir şekilde duracak ve karışıklıklar önlenecektir. İndirilen Linux çekirdeği sıkıştırılmış formattan "tar" komutu ile dizine çıkarılmalıdır. Terminalden "cd" komutu ile "mini6410/Linux-2.6.28" dizinine gelinerek "cp config_mini6410_a70" komutu çalıştırılarak konfigürasyon dosyası ilgili dizine kopyalanmalıdır. Burada mini6410 geliştirme kartı için 12 farklı konfigürasyon dosyası bulunmaktadır. Bu dosyaların farklılıkları sadece kullanılacak olan ekranların farklılığından kaynaklanmaktadır. Bu çalışmada kullanılan ekran için "config_mini6410_a70" konfigürasyon dosyası mevcuttur. Yukarıdaki dosya ile çekirdek derlenmesi için konfigürasyon ayarları yapılmaktadır. Bu ayarları görmek, düzenleme yapmak için "make menuconfig" komutu kullanılmalıdır. Bu komut çalıştırıldığında terminal penceresinde kullanıcı arayüzü açılacaktır. Bu işlemi yapmak için alternatif arayüzler mevcuttur. Ancak en kullanışlı olanı "menuconfig" arayüzü ile ayarlar menüsüne giriş yapmaktır. Bu çalışmada mini6410 geliştirme kartı SD kart üzerinden çalıştırılacağı için konfigürasyon ayarlarında sırasıyla "General setup" menüsüne girilmiştir. Daha sonra "Initial RAM filesystem and RAM disk (initramfs/initrd) support" ayarını "n" tuşu ile pasif yapılmalıdır. Daha sonra "esc" tuşuna iki defa basarak bir üst menüye daha sonra tekrar "esc" tuşuna iki defa basarak ayarlar menüsünden çıkış yapılmalıdır. Çıkışta yapılan değişiklikler kaydedilmiştir. Bu işlemler sonunda "Linux-2.6.28" dizininde ".config" dosyasının oluştuğu görülecektir.

Linux sistemlerinde nokta ile başlayan dosyalar gizli özellikte olduğu için “ctrl+h” komutu ile gizli dosyalar gösterilebilir. Menü ayarlarından istenilen değişiklikler yapılabilir, ancak konfigürasyon dosyası üzerinde fazla bir değişiklik yapılmaması sistemin kararlı çalışması için önemlidir. Çünkü birbiri ile bağımlı birçok özellik bulunmaktadır. Tüm bu işlemler sonunda çekirdek derlenmek için hazır olacaktır. Bulunulan dizini değiştirmeden “make” komutu ile çekirdek derleme işlemi başlatılabilir. Burada “make” komutuna ilaveten –j* komutu kullanılabilir. * yerine sistemde bulunan işlemci çekirdek sayısı yazılmalıdır. Böylece derleme süresi daha da kısaltılabilir. “j” parametresi kendisinden sonra thread sayısı olarak giriş almaktadır. Thread parametresi fazla artırılmamalıdır. Aksi takdirde hızlanmaktan öte yavaş derlemeye sebep olabilir. Çünkü derleme işleminde bağımlı işlemler çoğunluktadır. En ideal olan sayı sistemin çekirdek sayısını yazmaktır. Derleme işlemi bittikten sonra “Linux-2.6.28/arch/arm/boot” dizini içerisinde “zImage” dosyasının oluştuğu görülecektir. Bu dosya sistem için kullanılacak Linux çekirdeğidir.

EK C. Mini6410 Geliştirme kartı kök dosya sistemi oluşturma

Kök dosya sistemi oluşturmadan önce sisteme “debootstrap” programının kurulması gerekmektedir. Terminal üzerinden “sudo apt-get install debootstrap” komutu ile program yüklenir. Bu tez çalışmasında debian tabanlı bir Linux kök sistemi kullanılmıştır. “debootstrap” programı kullanılarak en son güncellenen kök dosya sistemi internet üzerinden indirilebilir. Bu işlem için “debootstrap --arch=armel --foreign squeeze rootfs/ http://ftp.us.debian.org/debian” komutu ile gerekli tüm dosyalar indirilir. Bu komutta armel ARM işlemciyi, “rootfs/” dosyaların indirileceği dizini ifade etmektedir. Bu işlem sonunda terminal penceresinin çalıştığı dizinde “rootfs/” dizinin oluştuğu ve içerisinde Linux sistem için gerekli dosyaların bulunduğu gözlemlenebilir (Şekil C.1).

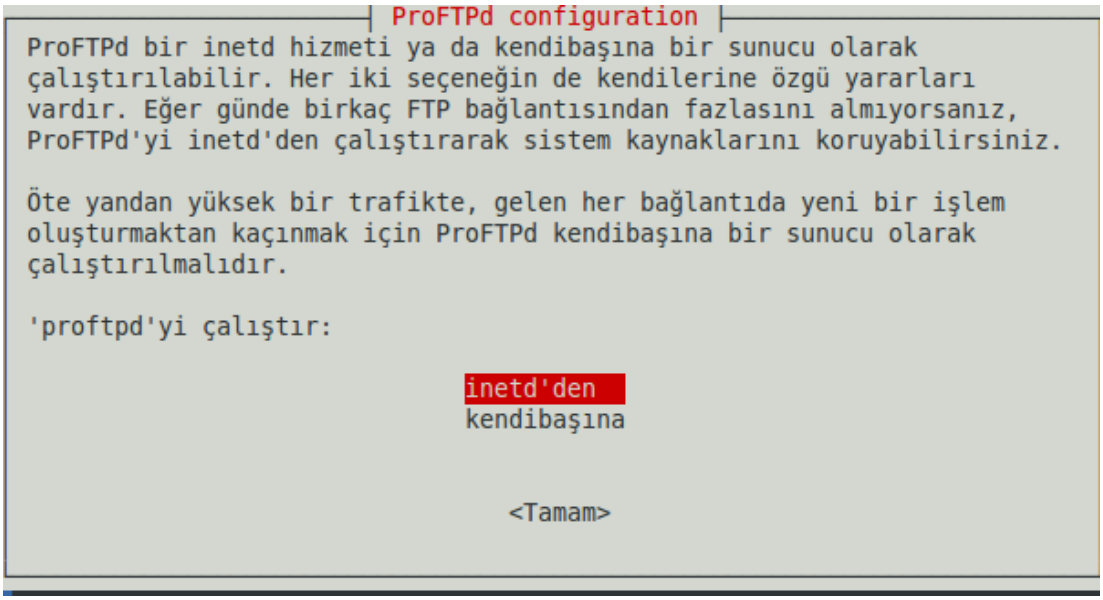


Şekil C.1. rootfs/ kök dosya sistemi

“rootfs/” dizini içinde bulunan tüm içerik mini6410 geliştirme kartında kullanılacak SD karta kopyalanmalıdır. Ayrıca internet üzerinden güncelleme yapılacak olan internet adresini kök dizinde ilgili dosyaya eklenmesi gerekmektedir. Debian tabanlı sistemlerde güncelleme adresinin tutulduğu konum “/etc/apt/sources.list” dosyasıdır. Bu dosya açılarak “deb http://ftp.us.debian.org/debian squeeze main” metni en başta bulunan satırla değiştirilmesi gerekmektedir. Bu işlemi komut satırı üzerinden “echo "deb http://ftp.us.debian.org/debian squeeze main" >> rootfs/etc/apt/sources.list” komutu verilerek de yapılabilir.

EK D. Linux tabanlı sistemlere FTP ve SSH kurulumu

Linux tabanlı bilgisayarlara FTP kurulumunu gerçekleştirmek için terminal penceresinden "sudo apt-get install proftpd" komutu verilmelidir. Komut çalıştıktan sonra Şekil D.1'deki gibi kullanıcıya soru soran bir pencere gelecektir.



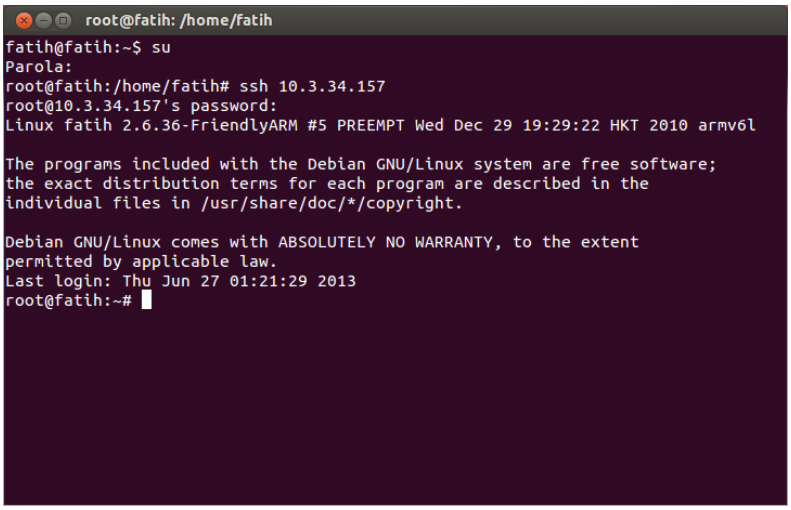
Şekil D.1. Proftpd kurulumu

Eğer aynı anda birden çok kullanıcı bağlantısı yapılacaksa "kendibaşına" seçeneği ile devam edilmelidir. Aksi durumda inetd'den (Sistem servisi) seçeneği ile kuruluma devam edilmelidir. Proftpd programının tüm ayarları "/etc/proftpd/proftpd.conf" dosyasında tutulmaktadır. Varsayılan ayarlar da bir değişiklik yapmadan sistemin kullanıcı adı ve parolası kullanılarak bağlantı gerçekleştirilebilir. Proftpd programı hem masaüstü bilgisayarımıza hem de mini6410 geliştirme kartına kurulması gerekmektedir.

FTP programı kurulduktan sonra dosya aktarım işlemi için farklı yöntemler kullanılabilir. Terminal penceresinden "ftp IP_adresi" yazılarak bağlantı kurulabilir. Ancak en kullanışlı yöntem FileZilla programını kullanmaktadır. FileZilla programı ile mini6410'un IP adresi kullanıcı adı ve parolası ile bağlantı yapılabilir. Mini6410 geliştirme kartı için yönetici kullanıcı adı "root"'dur. İlk kurulumda mini6410'a

yönetici şifresi atamak için “root password “****”” komutu ile istenilen şifre verilebilir. Burada verilen şifre FTP bağlantılarında kullanılacak olan şifredir. Şifre belirlendikten sonra FileZilla programı ile bağlantı gerçekleştirip dosya transferleri yapılabilir. Burada dikkat edilmesi gereken en önemli nokta FileZilla programında “Aktarım Tipi” olarak “ikili” seçeneği seçilmelidir. Aksi durumda mini6410’a yüklenen dosya çalıştırıldığı zaman “SEGMENTATION FAULT” hatası ile karşılaşılacaktır. Bir diğer önemli nokta ise mini6410’a yüklenen her dosya için çalıştırmadan önce “chmod 777 dosya_adi” komutu ile tam izin verilemesi gerektiğidir. Varsayılan olan olarak FTP ile gönderilen dosyalar kısıtlı izinlere sahip olduğundan çalıştırılamazlar.

Mini6410 geliştirme kartına klavye bağlayarak terminal ile istenilen tüm işlemler yapılabilir ancak bu her zaman pratik olmamaktadır. Bu nedenle mini6410’a internet üzerinden bağlanarak terminal komutlarını bilgisayardan yazmak çok daha pratik olacaktır. Bu işlemi terminal üzerinden FTP komutu ile bağlanılarak yapılabilir. Bu tez çalışmasında kullanımının çok pratik olması sebebiyle SSH protokolü kullanılmıştır. SSH, uzaktan başka bir cihaza bağlantı yapmaya yarayan FTP benzeri bir programdır. Terminal penceresini açarak “sudo apt-get install openssh-server” komutu ile SSH sunucu programını hem bilgisayara hem de mini6410 geliştirme kartına kurulması gerekmektedir. SSH programı kurulduktan sonra bilgisayar terminal penceresinden “SSH IP_adresi” komutu ile mini6410 geliştirme kartına bağlantı yapılabilir. Şekil D.2’de görüldüğü gibi mini6410 önce kullanıcı adı sonrada parola isteyecek ve doğrulama işlemi sonrasında aynen mini6410’da olduğu gibi terminal penceresi bilgisayarda görülecektir. Bu işlemden sonra mini6410’a bağlı klavyeden işlem yapıyor gibi tüm işlemler rahatlıkla yapılabilir. SSH bağlantısını kesmek için “exit” komutu ile çıkış yapılabilir.



Şekil D.2. SSH giriş ekranı

EK E. Temel Linux komutları

Bu başlık altında temel Linux komutları açıklanacaktır.

man <komut> : man komutu, parametre olarak verilen komut hakkında bilgiler vermektedir.
<komut> --help : komutu, verilen komutun kullanılan ayarlarını göstermektedir.
mkdir dizin_adi : Verilen isimde, çalışılan dizinde, yeni bir dizin oluşturur.
ls : Terminalde bulunulan dizindeki tüm dosya ve klasörleri listeler.
ls -l : Tüm dosya ve klasörleri, oluşturulma tarihleri ve izinleri ile birlikte listeler.
cd [dizin] : Çalışılan dizini değiştirerek verilen dizine geçiş yapar.
pwd : Çalışılan dizinin tam yolunu gösterir.
cp kaynak, hedef : Bu komut kaynak dosyayı hedef konumuna kopyalar.
mv eski_ad, yeni_ad : Bu komut, dosya isimlerinde değişiklik yapar. Aynı zamanda hedef olarak dizin gösterilirse, belirtilen dosyayı taşıma işlemi yapar.
rm dosya / rm –rf dizin : Belirtilen dosya yada dizini siler. Dizin silmek için –rf ayarı ile kullanılmalıdır.
find -name, dosya : Belirtilen dosyayı çalışma dizininde arar. Bulunan sonuçları listeler.
history : Son kullanılan komutları sondan başa doğru listeler.
echo [metin] : Verilen metnin ilk satırını ekrana yazar.
grep pattern : Bu komut, verilen pattern'i dizinde arar, hangi satırda geçtiğini gösterir.
sort dosya.txt : Metin dosyası içinde bulunan satırları alfabetik olarak listeler. –r parametresi ile tam tersi olarak listeler.
chmod dosya / chmod –R dizin : Dosya yada dizin izinlerini düzenler.
chown : Dosya yada dizinin dahil olduğu grubu ve sahibini değiştirir.
su : Terminal kullanıcısını yönetici olarak değiştirir.
sudo <komut> : Çalıştırılacak komutu yönetici olarak çalıştırır.
passwd kullanıcı_adı : Verilen kullanıcının parolasını değiştirir.
who : Terminalde çalışan kullanıcıyı görüntüler.
tar *.tar, hedef: Sıkıştırılmış dosyayı belirtilen dizine çıkartır. Aynı zamanda sıkıştırma işlemi yapar.
ssh IP : Uzaktan başka bir cihaza bağlantı yapar.

reboot : Sistemi yeniden başlatır.

poweroff : Sistemi kapatır.

make : Derleme işlemini başlatır.

make install : Derleme işlemi sonucunda oluşması gereken dosyaları oluşturur.

configure : Derleme işlemi için MakeFile dosyası oluşturur.

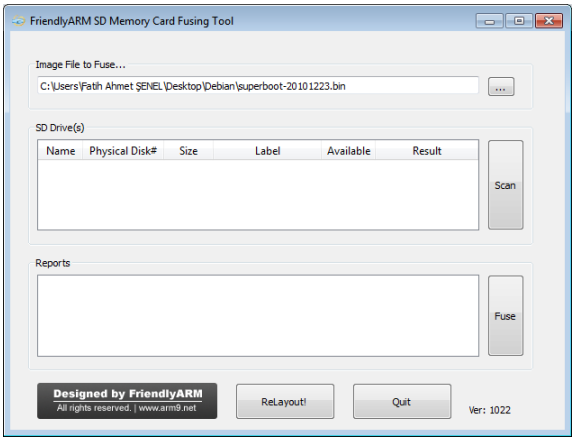
EK F. Mini6410 geliştirme kartı özellikleri

- **Dimension:** 110 x 110 mm
- **CPU:** 533 MHz Samsung S3C6410A ARM1176JZF-S with VFP-Unit and Jazelle (max freq. 667 MHz)
- **RAM:** up to 256 MB DDR RAM, 32 bit Bus
- **Flash:** up to 1GB NAND Flash
- **EEPROM:** 256 Byte (I2C)
- **Ext. Memory:** SD-Card socket
- **Serial Ports:** 1x DB9 connector (RS232), total: 4x serial port connectors
- **IR:** Infrared Receiver
- **USB:** 1x USB-A Host 1.1, 1x miniUSB Slave/OTG 2.0
- **Audio Output:** 3.5 mm stereo jack
- **Audio Input:** Condenser microphone
- **Ethernet:** RJ-45 10/100M (DM9000)
- **RTC:** Real Time Clock with battery
- **Beeper:** PWM buzzer
- **Camera:** 20 pin (2.0 mm) Camera interface
- **TV Output:** AV Out
- **LCD Interface:** 40 pin (2.0 mm) and 41 pin connector for FriendlyARM Displays and VGA Board
- **User Inputs:** 8x push buttons and 1x A/D pot
- **User Outputs:** 4x LEDs
- **Expansion:** 40 pin System Bus, 30 pin GPIO, 20 pin SDIO (SD, SPI, I2C), 10 pin Buttons (2.0 mm)
- **Debug:** 10 pin JTAG (2.0 mm)
- **Power:** regulated 5V
- **Power Consumption:** Mini6410: 0.25 A, Mini6410 + 4.3" LCD: 0.5 A
- **OS Support**
    - Windows CE 6
    - Linux
    - Android
    - Ubuntu

EK G. Mini6410 gömülü Linux kurulumu

Mini6410 geliştirme kartı SD kart ve USB üzerinden kurulum desteklemektedir. Bu tez çalışması SD üzerinden kurulum ile gerçekleştirildiğinden USB ile kurulum anlatılmayacaktır. SD kart ile kurulum işlemini yapmadan önce SD kart üzerinde birkaç işlem yapmak gerekmektedir. ARM işlemcinin açılış anında SD kartı okuyarak yüklemeyi başlatabilmesi için SD kartın bootable özelliğinin olması gerekmektedir.

SD kart kullanılmadan önce "SD-Flasher" programı ile Windows ortamında mini6410 için boot edilebilir özellik kazandırılması gerekir. Bu işlemi yapmak için öncelikle SD kart, kart okuyucuya yerleştirilir ve program çalıştırılır. Bu işlemden önce SD kartta varsa verilerin yedeklenmesi gerekmektedir aksi halde tüm veriler kaybolacaktır. Mini6410 geliştirme kartının geliştiricisi olan FriendlyARM firmasının internet sitesinden en güncel "superboot.bin" dosyası indirilir. Daha sonra Şekil G.1'de görüldüğü gibi SD-Flasher programı çalıştırılır.



Şekil G.1. SD-Flasher programı görüntüsü

"Image File to Fuse…" yoluna internet sitesinden indirilen "superboot.bin" dosyası gösterilir. "Scan" butonu ile sisteme bağlı olan SD kartların listesi alınır. İşlem yapılacak olan SD kart seçildikten sonra "Fuse" butonu ile işlem başlatılır. Eğer Windows7 işletim sistemi ile çalışılıyorsa "Fuse" butonundan önce "ReLayout" butonu ile bir ön işlem yapılması gerekecektir. Son olarak kullanıma hazır hale gelen

SD kart için kök dizine “images” adında bir klasör oluşturarak SD kart kullanıma hazır hale getirilir.

Mini6410 geliştirme kartının hangi frekans değerinde çalışacağı işletim sistemi kurulumundan önce ayarlanmalıdır. Bu işlem U-boot işlemi ile gerçekleştirilmektedir. Aynı zamanda U-boot işlemi kullanılacak RAM bellek miktarının da ayarlanabildiği aşamadır.

U-boot, ARM işlemcisinin kendi içine, boot işlemi sırasında yapması gereken işlemlerin kodlandığı işlemdir. Yani işlemciye enerji verildiğinde ne yapacağı, hangi bellek adresinden hangi bilgiyi okuyacağı, işlemcinin hangi frekansta çalışacağı gibi işlemler bu aşamada yapılmaktadır. Mini6410 geliştirme kartına ait U-boot dosyası geliştirici firmanın internet sitesinden indirilebilmektedir. İndirilen dosya sıkıştırılmış formattan, istenilen dizine çıkarılıp “make” ve “make install” komutu ile doğrudan derlenip kullanılabilir. Ancak değişiklik yapılması gerekiyorsa derleme işleminden önce, istenilen değişiklikler yapılmalıdır. Bu tez çalışmasında 533Mhz-667Mhz frekanslarında çalışabilen S3C6410 işlemcisi için frekans ayarlaması yapılmıştır. Varsayılan olarak bu değer 533Mhz değerine ayarlanmıştır. Bu işlemi yapmak için “/u-boot/include/configs/mini6410.h” dosyasında değişiklik yapılması gerekmektedir. Dosya açılarak “#define APLL_MDIV ***” yazan bölüm bulunup, “***” yazan yere işlemcinin çalışmasını istediğimiz frekans değerinin yarısı yazılmalıdır. Bu dosya incelendiğinde “APLL_MDIV” değerinin başka yerlerde kullanıldığı ve işlemci frekans değeri hesaplanırken iki katının alındığı görülebilir. Bu nedenle çalışmasını istediğimiz frekans değerinin yarısı yazılmalıdır. Bu değer atama yapılırken 533-667 frekans değerleri arası tercih edilmelidir. 667Mhz değerinden büyük değerlerde de işlemci çalışmaktadır. Ancak işlemci daha fazla ısınmakta ve çok hızlı frekans değerlerinde boot işlemi yaparken, boot dosyası işlemci tarafından görülmeyebilmektedir. Bu nedenle ideal aralıkta tercih yapılmalıdır. Bu tez çalışmasında “APLL_MDIV” değişkenine 333 değeri atanarak 666Mhz değerinde işlemci çalıştırılmıştır. Dosyada gerekli değişiklikler yapıldıktan sonra kaydedilip, terminal üzerinden derleme işlemi yapılmalıdır. “make” komutu ile derlemeyi başlatmadan önce, U-boot’un sistemi nasıl kullanacağı ile ilgili olarak son bir ayar yapmak gerekmektedir. “make mini6410_nand_config-ram256” komutu ile sistemin NAND belleğine U-boot’un yazılacağı ve sistem çalıştığı sürece 256 MB

RAM bellek kullanacağı bildirilmektedir. Mini6410, 256 MB RAM belleğe sahip ancak istenildiğinde 128 MB RAM kullanacak şekilde de ayarlanabilmektedir. "MakeFile" dosyası incelendiğinde işlemci tarafından desteklenen tüm ayarlar incelenebilir. Bu işlemden sonra "make" ve "make install" komutu ile derleme işlemi yapılıp "u-boot.bin" dosyası elde edilmektedir.

Mini6410 geliştirme kartı, hazırlanan SD karta gerekli gömülü Linux dizinleri kopyalandıktan sonra kullanılmaya başlanabilir.

SD karttan boot etmek yerine gerekli işletim sistemi dosyalarını kartın NAND flash belleğine kopyalamak suretiyle kartı SD kart olmadan daha hızlı bir şekilde açılıp kullanma imkânı vardır. Bu işlem için SD karttan boot etme başlığında anlatılan işlemler aynen uygulanır. Tek fark FriendlyArm.ini dosyasındaki küçük değişikliklerdir. Şekil G.2'de görüldüğü gibi Action parametresine "Install" dediğimizde sistem SD karttan başlamak yerine gerekli dosyaları belleğe kopyalayarak sistemin kuruluşunu gerçekleştirecektir.

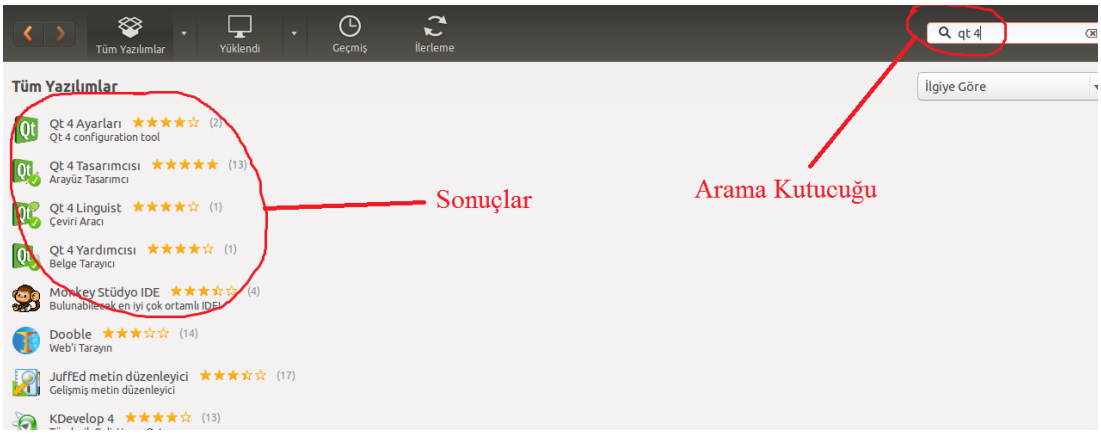
```
#This line cannot be removed. by FriendlyARM(www.arm9.net)

CheckOneButton=No
Action=install
OS= linux

VerifyNandWrite=No
StatusType = Beeper| LED

###################### Linux ######################
Linux-BootLoader = Linux/u-boot.bin
Linux-Kernel = Linux/zImage
Linux-CommandLine = root=/dev/mmcblk0p1 noinitrd console=ttySAC0,
115200 init=/bin/sh
```

Şekil G.2. FriendlyARM Dosyası

SD kart içinde dosyaların doğru bir şekilde yerleşmemesi ya da FriendlyArm.ini dosyasının yanlış doldurulması sonucunda boot işlemi başlamayacaktır. Boot işlemi başarılı bir kilde başladığında mini6410 geliştirme kartı bir adet bip sesi çıkaracaktır ve boot işlemi başlayacaktır. Ses duyulmadığı zaman dosyaların yanlış yerleştiği ya da FriendlyArm.ini dosyasının yanlış doldurulduğu anlaşılmalıdır. Bu durumda tüm işlemler baştan doğru bir şekilde tekrarlanmalıdır. Bu işlem ilk yapıldığında dosyaların kopyalanması birkaç dakika kadar sürecektir. Ancak daha sonraki açılışlarda sistem hızlıca kendini NAND Flash'tan boot edecek ve açılacaktır. Kurulum işleminin doğru gittiği kart üzerinde bulunan dört adet kullanıcı LED'lerinden takip edilebilir. LED'ler ilk başta birincisi yanıp sönecek, daha sonra birinci LED sabit yanarken ikinci LED yanıp sönecek ve bu işlem dördüncü LED yanıp sönme işlemi bittiğinde tamamlanacaktır. Sistem kurulumu tamamlandığında geliştirme kartından iki adet kısa aralıklarla bip sesi duyulacaktır. Daha sonra kartı SD kartı çıkardıktan sonra tekrar başlatıldığında mini6410 kullanıma hazır hale gelecektir.

EK H. Qt kurulumu ve ayarlarının yapılması

Ubuntu işletim sistemine herhangi bir yazılım kurmak oldukça basittir. Bu işlem için çoğunlukla Ubuntu içinde hazır olarak bulunan "Ubuntu Yazılım Merkezi" kullanılmaktadır. Şekil H.1'de görüldüğü gibi yazılım merkezi açıldıktan sonra yüklemek istenilen program arama kutucuğuna yazılarak aratılır. Daha sonra seçip yükle denildiğinde sistem parolası sorulmakta ve kurulum işlemi başlamaktadır. Yazılım merkezinde aratıp bulanamayan yazılımların kurulum dosyalarını internetten indirerek terminal üzerinden kurulumu gerçekleştirilebilir. Yazılım merkezi üzerinden yapılan tüm işlemler terminal üzerinden de yapılabilir ancak daha zahmetli olacaktır.



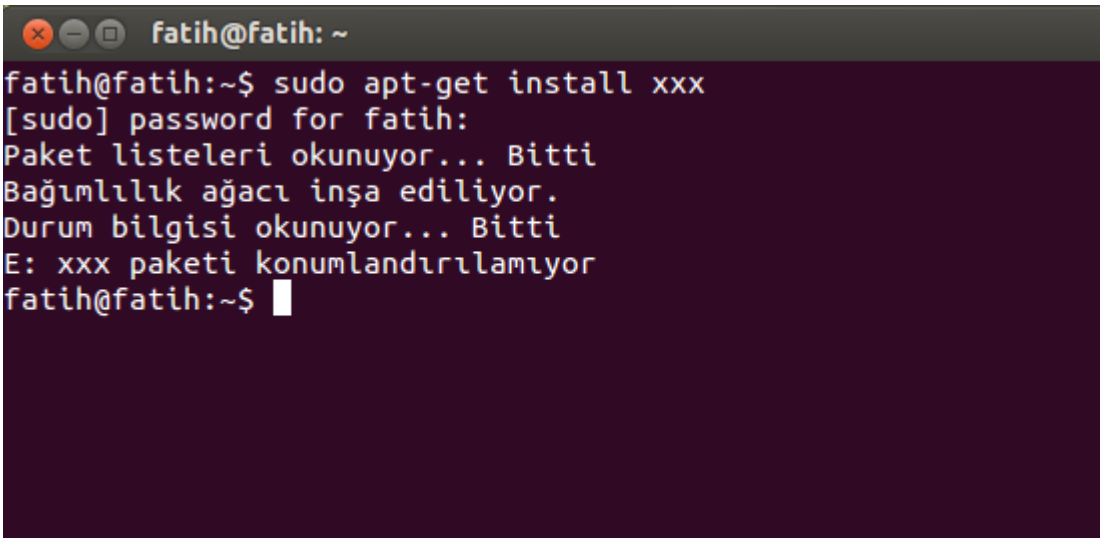
Şekil H.1. Ubuntu yazılım merkezi

Yazılım merkezi kullanıcının yerine yapılması gereken tüm işlemleri arka planda yapmaktadır. Komut satırından sırasıyla aşağıdaki komutlar verilerek de kurulum işlemi yapılabilir;

```
sudo apt-get install libqwt5-qt4-dev
sudo apt-get install libqwt5-doc
sudo apt-get install qtcreator
```

Kurulum işlemi ile birlikte öncelikle Qt için gerekli kütüphaneler kurulmakta ve sonrasında Qt kurulumu başlatılmaktadır. Bu işlem sonunda Ubuntu'ya Qt'nin en yeni versiyonu kurulmuş olacaktır. Tabi burada anlatılanlar Ubuntu tarafından

bilinen dosyaların kurulumu içindir. Örneğin sistemin tanımadığı "xxx" adında bir programı yüklemek istediğimizde Şekil H.2'de görüldüğü gibi "E: xxx paketi konumlandırılamıyor" hata mesajı ile karşılaşılır. Çünkü "xxx" programı sistem tarafından bilinmemektedir.



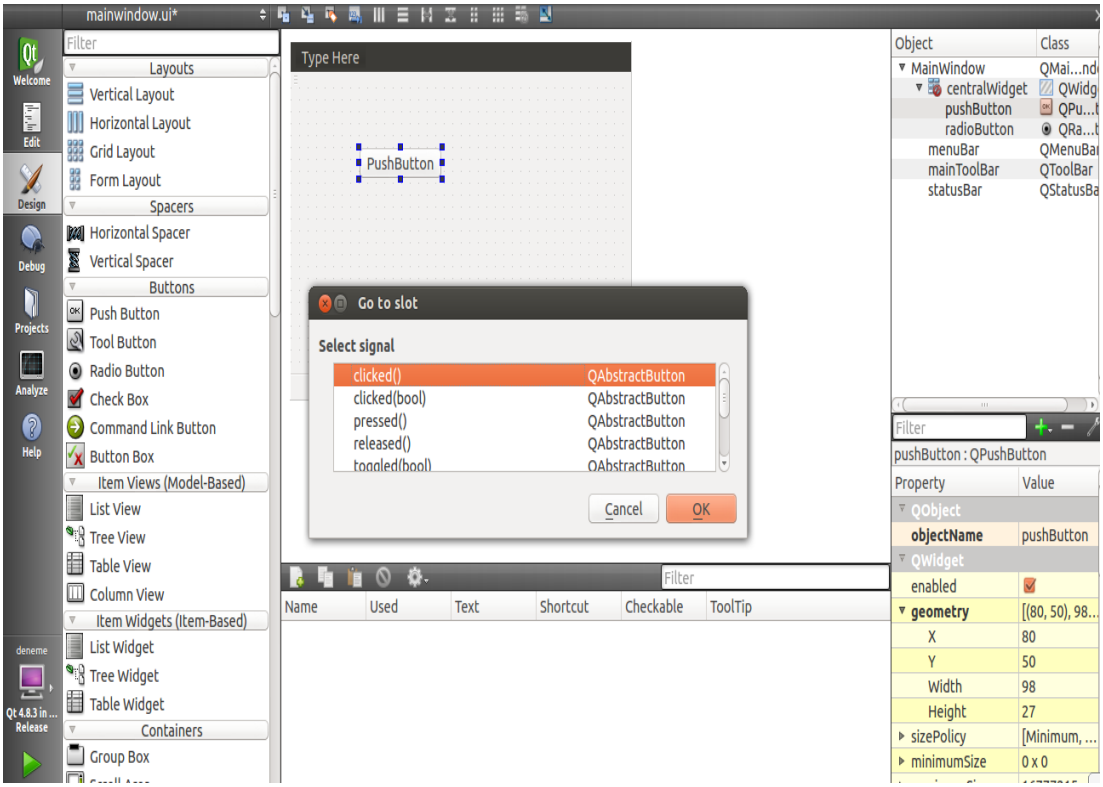
Şekil H.2. Bilinmeyen uygulama hata mesajı

Sistem tarafından tanınmayan kurulum dosyaları *.sh, *.bin ve *.deb uzantılarından birine sahip olmalıdır. Terminal penceresi açılarak kurulum yapılacak dosya ile aynı dizine gelinip kurulum yapılacak dosyaya öncelikli olarak izin verilmelidir. Daha sonra;

*.sh için; "sh xxx.sh"
*.bin için; "./xxx.bin"
*.deb için; "sudo dpkg -i xxx.deb" komutları verilerek kurulum işlemi başlatılmalıdır. Kurulum için gerekiyorsa, internetten dosyayı terminal indirecek ve yükleme işlemini tamamlayacaktır. Kurulum esnasında kullanıcıdan, hafızada kullanılacak yerler hakkında onay istenebilmektedir. Onay verilerek kurulum işlemi tamamlanır.

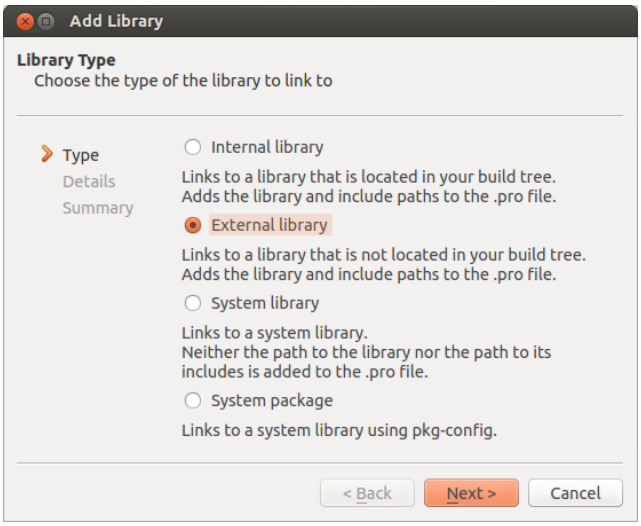
Bu tez çalışmasında kullanıcı arayüz formları tasarlama imkânı sağladığından dolayı Qt geliştirme ortamı tercih edilmiştir. Qt sürükle bırak özelliği ile kullanıcı arabirimi oluşturma işlemi çok basittir. İstenen araçlar Şekil H.3'de görülen sol taraftaki

toolbox'tan seçilip forma sürüklenerek eklenmektedir. Eklenen araca sağ tıklama yapılarak "Goto Slot" seçeneği ile araca ait tüm event'ler listelenmektedir. İstenilen event seçilerek doğrudan kod bölümüne geçilebilir.



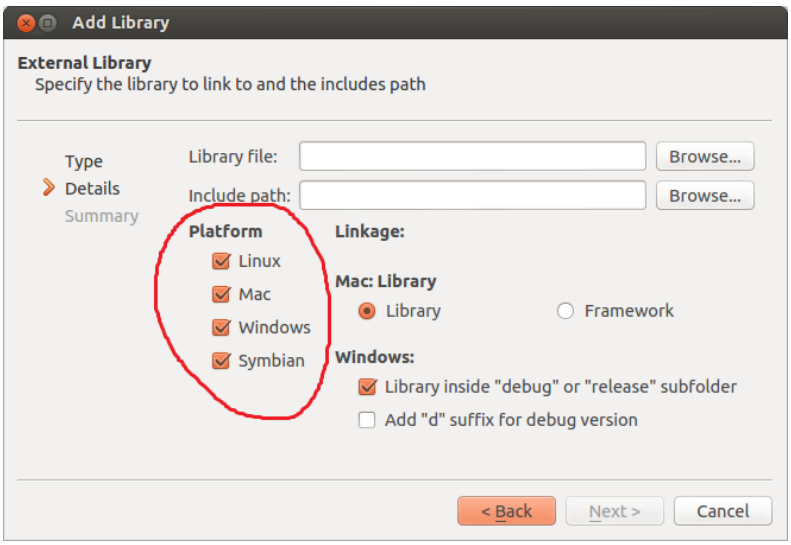
Şekil H.3. Qt GUI geliştirme arayüzü

Qt projelerinde ayarların yapıldığı, kütüphane dosyalarının eklendiği dosya Qt proje (.pro) dosyasıdır. Bu tez çalışmasında OpenCV görüntü işleme kütüphaneleri ve seri haberleşme kütüphaneleri kullanılmadan önce proje dosyasına eklenmiştir. Proje dosyasında boş herhangi bir yere farenin sağ tuşu ile tıklandığında "Add Library" ve "External Library" seçeneği ile kütüphane eklenebilir (Şekil H.4).



Şekil H.4. Harici kütüphane ekleme

Şekil H.5'te görüldüğü gibi sadece çalışılacak olan işletim sistemi seçilerek gereksiz eklentileri eklemekten kaçınılmalıdır.



Şekil H.5. Platform ve kütüphane seçimi

Daha sonra Header ve Include ile belirtilen yerlere eklenecek dosyaların yolu gösterilerek ekleme işlemi gerçekleştirilir. Header yolu seçildiği zaman, Qt otomatik olarak Include yolunu seçecektir. Bu tez çalışmasında geliştirilen kod öncelikle kişisel bilgisayarda denenmiş daha sonra çapraz derleme ile geliştirme kartına göre derlenmiştir. Burada dikkat edilmesi gereken nokta, OpenCV görüntü işleme kütüphanesi kullanılırken, kişisel bilgisayarda kullanılan ile geliştirme kartı için

kullanılan OpenCV kütüphaneleri farklıdır. Bu nedenle geliştirme kartı için çapraz derleme aşamasına geçmeden önce proje dosyasında (.pro) eklenilen OpenCV kütüphanelerini, ARM geliştirme kartı için olan kütüphanelerin yolunu vererek güncellemek gerekmektedir. Aksi takdirde çapraz derleme aracı OpenCV kütüphanelerini okuyamayacak ve derleme sırasında hata verecektir. Bir diğer önemli nokta ise OpenCV kütüphanelerinin proje dosyasına eklenme sırası kişisel bilgisayarlarda herhangi bir sorun oluşturmazken, çapraz derleme sırasında özel bir sırada olmadığı takdirde hata vermektedir. Kütüphane eklenme sırası Şekil H.6'da gösterilmiştir.

```
unix:!macx:!symbian: LIBS += -L$$PWD/../240-180-install-2.4/lib/ -lopencv_highgui

INCLUDEPATH += $$PWD/../240-180-install-2.4/include
DEPENDPATH += $$PWD/../240-180-install-2.4/include

unix:!macx:!symbian: PRE_TARGETDEPS += $$PWD/../240-180-install-2.4/lib/libopencv_highgui.a


unix:!macx:!symbian: LIBS += -L$$PWD/../240-180-install-2.4/lib/ -lopencv_imgproc

INCLUDEPATH += $$PWD/../240-180-install-2.4/include
DEPENDPATH += $$PWD/../240-180-install-2.4/include

unix:!macx:!symbian: PRE_TARGETDEPS += $$PWD/../240-180-install-2.4/lib/libopencv_imgproc.a

unix:!macx:!symbian: LIBS += -L$$PWD/../240-180-install-2.4/lib/ -lopencv_core

INCLUDEPATH += $$PWD/../240-180-install-2.4/include
DEPENDPATH += $$PWD/../240-180-install-2.4/include

unix:!macx:!symbian: PRE_TARGETDEPS += $$PWD/../240-180-install-2.4/lib/libopencv_core.a


unix:!macx:!symbian: LIBS += -L$$PWD/../240-180-install-2.4/lib/ -lopencv_calib3d

INCLUDEPATH += $$PWD/../240-180-install-2.4/include
DEPENDPATH += $$PWD/../240-180-install-2.4/include

unix:!macx:!symbian: PRE_TARGETDEPS += $$PWD/../240-180-install-2.4/lib/libopencv_calib3d.a

unix:!macx:!symbian: LIBS += -L$$PWD/../240-180-install-2.4/share/OpenCV/3rdparty/lib/ -llibjasper

INCLUDEPATH += $$PWD/../240-180-install-2.4/share/OpenCV/3rdparty
DEPENDPATH += $$PWD/../240-180-install-2.4/share/OpenCV/3rdparty
```

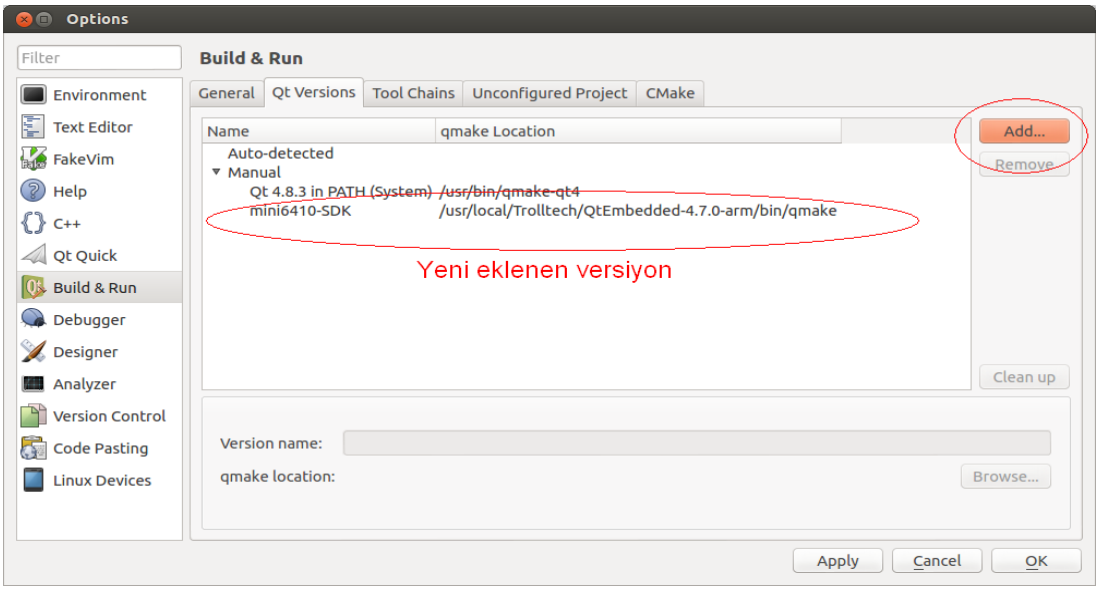
Şekil H.6. OpenCV kütüphane sırası

Bu sıralamadan sonra geriye kalan kütüphaneler için her hangi bir sıra takip edilmesi önemli değildir. Ancak yukarıda belirtilen temel kütüphaneler birbirine bağımlı çalıştığından dolayı sırası önemlidir. Geliştirme kartı için OpenCV kütüphanelerini derleme işlemi Ek I'da anlatılmıştır.

Çapraz derleme araçları üçüncü bölümde ayrıntılı olarak anlatılmıştır. Bu başlık altında Qt geliştirme ortamına çapraz derleme araçlarının nasıl eklendiği ve nasıl kullanıldığı anlatılacaktır.

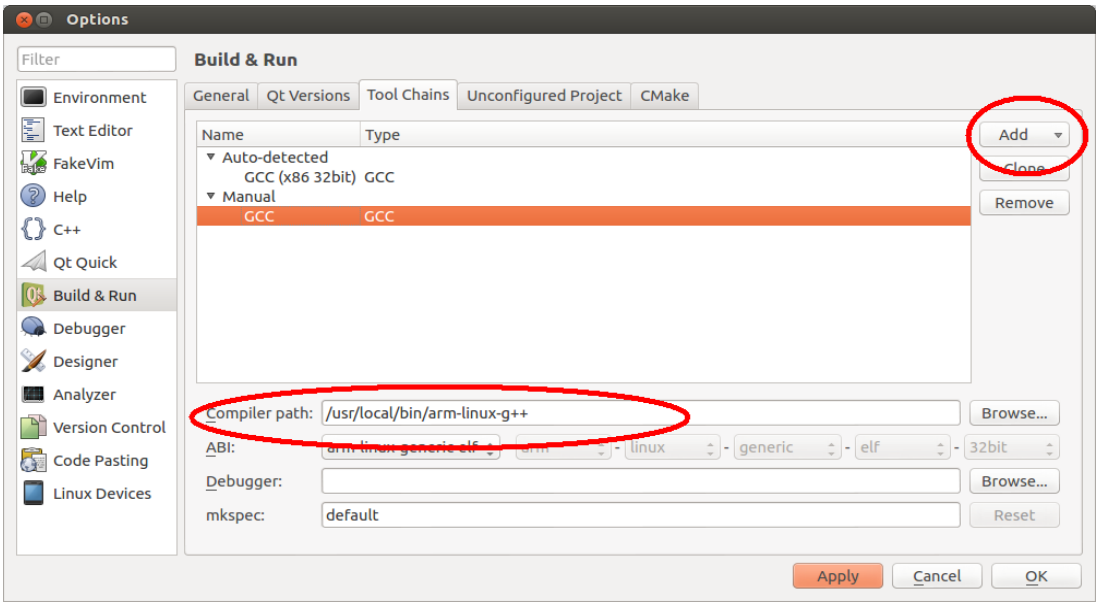
Öncelikli olarak Qt geliştirme ortamı, bilgisayara kurulduktan sonra ARM işlemciler için olan versiyonunun da sisteme kurulması gerekmektedir. Qt geliştiricileri tarafından ARM tabanlı gömülü sistemler için geliştirilmiş ücretsiz olarak dağıtılan birden fazla versiyonlar mevcuttur. Qt geliştirme sitesinden veya ARM geliştirme kartının üreticisi olan firmanın internet sitesinden bu versiyon indirilip sisteme kurulabilir. İndirilen dosya içinde kurulum işlemi için gerekli olan tüm komutlar sırasıyla "build-all" adında bir dosyada hazırlanmıştır. Linux sistemlerde "./" operatörü ile içerisinde komutlar bulunan dosyalar çalıştırılabilmektedir. Kurulum dosyasının sıkıştırılmış formattan istenilen bir yere çıkartıp komut satırına yönetici yetkisi ile giriş yaparak ("su" komutu ile şifre girilerek) "./build-all" komutu çalıştırıldığında sistem gerekli olan tüm kurulumları yapacaktır. Bu işlem bilgisayarın performansına göre uzun sürebilmektedir. Bu tez çalışmasında ARM işlemciler için Qt 4.7.0 versiyonu kullanılmıştır. Kurulum tamamlandığında "/usr/local/Trolltech/QtEmbedded-4.7.0-arm" dizinine gerekli olan tüm dosyalar kopyalanmış olacaktır. Bu işlemi yapmadan önce ARM işlemciler için gerekli olan çapraz derleme araçlarının sisteme kurulu olduğundan emin olunmalıdır. Aksi takdirde terminal penceresinde hatalar ile karşılaşılacaktır. "build-all" dosyası incelenerek değişiklikler yapılabilir. Örneğin; kurulum dizini istenilen başka bir yer ile değiştirilebilir. Kullanılacak çapraz derleme araçları düzenlenebilir. Ancak varsayılan değerler hatasız bir şekilde kurulum işleminin tamamlanması için uygun değerlerdedir.

Qt Creator programına ilk olarak Qt Embedded derleme aracının tanıtılması gerekmektedir. Bu işlem için Qt programını açtıktan sonra "Tools", "Options" menüsünü takip ederek ayarlar menüsüne giriş yapılır. Şekil H.7'de görüldüğü gibi Build & Run sekmesinden Qt Versions sekmesinden Add diyerek gömülü sistemler için Qt dizin yolu belirtilip programa tanıtılabilir. Varsayılan değerler ile kurulum yapıldığında gömülü sistemler için Qt derleme aracı "/usr/local/Trolltech/QtEmbedded-4.7.0-arm/bin/qmake" yolunda bulunmaktadır. İstenilen bir derleme aracı ismi ve dizin yolu verilerek, Qt gömülü sistem derleme aracı Qt Creator programına tanıtılmış olur.



Şekil H.7. Qt gömülü sistem derleme aracı ekleme ekranı

Son olarak, eklenilen Qt gömülü sistemi için, çapraz derleme aracı eklenmesi gerekmektedir. Aynı menü penceresinde Tool Chains sekmesi açılarak ikinci bölümde çapraz derleme aracının yüklendiği dizin yolu verilerek çapraz derleme aracı da programa tanıtılmış olacaktır (Şekil H.8). Unconfigured Projects sekmesinde eklenilen Qt gömülü sistem derleme aracına karşılık, çapraz derleme aracını ilişkilendirip ayarlar penceresi kapatılabilir.



Şekil H.8. Qt gömülü sistem toolchain ekleme ekranı

EK I. OpenCV gömülü sistemler için kurulumu

CMAKE çapraz platformlar için açık kaynak kodlu bir derleme aracıdır. CMAKE aracı kullanılarak çalışılan sistemin haricinde, farklı bir işlemciye ve işletim sistemine sahip bir derleme işlemi yapılabilir. Özellikle gömülü sistemler üzerine çalışma yapılıyorsa mutlaka bu araç ya da benzeri farklı bir araç kullanılması çok pratik olacaktır. CMAKE aracının iki farklı kullanımı mevcuttur. Bunlardan ilki kullanıcıya bir arayüz sunan GUI versiyonudur. Diğeri kullanıcı arayüzü olmadan komut satırı üzerinden çalışan versiyondur. Komut satırı üzerinden çalışan versiyonu daha sağlıklı olduğundan bu tez çalışmasında CMAKE aracı ile yapılan tüm işlemler komut satırı üzerinden gerçekleştirilmiştir. Ubuntu üzerinde CMAKE aracını çalıştırabilmek için önce kurulması gerekmektedir. Ubuntu kurulu bir sisteme herhangi bir programı kurmak için iki tane alternatif vardır. Birincisi Ubuntu Yazılım Merkezini kullanarak kurulum, diğeri gerekli kurulum dosyalarını indirerek komut satırı üzerinden kurulum yapmaktır. Yazılım Merkezi kullanılarak hemen hemen gerekli tüm programlar kurulabilmektedir. Ancak nadirde olsa kurmak istenilen program Yazılım Merkezinde mevcut olmayabilir. Bu durumda komut satırı üzerinden kurulum dosyaları indirilerek kurulum işlemi yapılması gerekmektedir. Ubuntu çalıştıran sistemlerde herhangi bir kurulum işlemi yapabilmek için "sudo" komutu kullanılmalıdır. Komuttan önce verilen "sudo" komutu işlenecek olan komutun yönetici olarak çalıştırılacağını belirtmektedir. Komut penceresi ilk açıldığında, "sudo" komutu ile bir işlem yapmak istenildiğinde kullanıcıdan yönetici parolası sorulacaktır. Açılan komut penceresi kapatılmadığı sürece "sudo" komutu için bir defa parola girilmesi yeterlidir. Yeni pencere açıldığında ya da pencere kapatılıp tekrar açıldığında "sudo" komutu ile giriş yapıldığında tekrar parola soracaktır. Yazılan her komutun başına "sudo" eklemek yerine ilk pencere açıldığı zaman "su" komutu çalıştırıldığında pencere otomatik olarak parolayı soracak ve yönetici oturumuna geçecektir. Aşağıdaki resimlerde görüldüğü gibi "su" komutu öncesi ve sonrası komut satırında görünen kullanıcı isimleri farklıdır. Bir defa "su" komutu ile giriş yaptığımızda artık pencere kapatılmadığı sürece "sudo" komutu yazılmaksızın çalıştırılan tüm komutlar yönetici haklarına sahip olarak çalıştırılacaktır. Yönetici hesabından çıkmak için "exit" komutu verilerek tekrar eski kullanıcıya dönülebilir. "apt-get install" komutu Ubuntu sistemler için kurulum komutudur. Kurulum komutu yönetici olarak çalıştırmayı gerektirmektedir. CMAKE

aracını kurmak için komut penceresini açılır ve “sudo apt-get install CMAKE” komutu çalıştırılır. Kurulum işlemi gerekli olan dosyaları internet üzerinden otomatik olarak indirip kurmaya başlayacaktır. Bu esnada kullanıcıdan “Devam etmek istiyor musunuz?” benzeri sorular sorabilmektedir. Gerekli kurulum işlemi tamamlanınca CMAKE aracı kullanıma hazır hale gelmiş olacaktır. Bu tez çalışmasında OpenCV görüntü işleme kütüphanesi mini6410 geliştirme kartında çalıştırılacaktır. Bu nedenle ARM işlemci mimarisine göre derlenmesi gerekmektedir. Farklı alternatif derleme yöntemleri vardır. Bu tez çalışmasında CMAKE aracı ile kurulum en pratik yöntem olduğu için tercih edilmiştir.

OpenCV kütüphanesi statik ve dinamik olmak üzere iki farklı şekilde derlenebilir. Gömülü sistem için OpenCV kütüphaneleri derlenmeye başlamadan önce hangi tipte derleneceğine karar verilmelidir. Dinamik derleme işleminde OpenCV kütüphanelerini kullanan uygulamalar, derlendiği zaman, derleme çıktısı olan çalıştırılabilir dosyaya OpenCV kütüphanelerini de ekler. Böylece çalıştırılabilir dosyanın boyutu artmış olur. Dinamik derlemenin avantajı, uygulama geliştirildikten sonra çalıştırılacak olan platforma OpenCV kütüphanelerini kopyalama zorunluluğu olmamasıdır. Çünkü gerekli olan tüm kütüphaneler çalıştırılabilir dosya içine eklenmektedir.

Statik derleme işleminde ise OpenCV kütüphaneleri çalıştırılabilir dosyaya eklenmez. Dolayısıyla OpenCV kütüphaneleri uygulamanın çalıştırılacağı platforma kopyalanması gerekmektedir. Bu yöntemin avantajı çalıştırılabilir dosyanın boyutunun küçük olmasıdır.

Statik kütüphanelerin uzantıları “*.so”, dinamik kütüphanelerin uzantıları “*.a” olmaktadır. Böylece kütüphanelerin uzantılarına bakılarak hangi türde derlendiği anlaşılabilir.

OpenCV kütüphanesinin statik ya da dinamik olarak derleneceği, derleme esnasında belirtilmektedir.

OpenCV kütüphanesi internetten indirildikten sonra bilgisayarda uygun bir dizine sıkıştırılmış formattan çıkarılır. Daha sonra terminal penceresi açılarak OpenCV

dizinine gelinir. OpenCV içinde "cmake" dizininde "OpenCVCompilerOptions.cmake" dosyası bulunmaktadır. CMAKE aracı ile ilgili olarak tüm ayarlamalar bu dosya içinde tutulmaktadır. Bu dosya ile ilgili herhangi bir hata derleme esnasında oluşursa, belirtilen hata mesajı "OpenCVCompilerOptions.cmake" dosyasında aranarak bulunur ve hataya neden olan özellik tamamen dosyadan silinir. Bu hatanın sebebi, derlemeye çalıştığımız çapraz derleme aracının, yani işlemcinin, hataya neden olan özelliği desteklemiyor olmasıdır. Bu nedenle bu özellikler dosyadan silinerek derleme dışı bırakılmalıdır. Eğer hiçbir hata meydana gelmezse işlemcimiz dosyada bulunan tüm özellikleri destekliyor demektir.

OpenCV dizini için "build" adında bir dizin oluşturarak terminal üzerinden dizin içine girilir. Daha sonra aşağıda belirtilen komut terminal üzerinde çalıştırılır.

```
cmake \
-D WITH_FFMPEG=0 \
-D CMAKE_CXX_COMPILER=/usr/local/bin/arm-linux-g++ \
-D CMAKE_C_COMPILER=/usr/local/bin/arm-linux-gcc\
-D CMAKE_CXX_FLAGS=-I/usr/local/arm-none-linux-gnueabi/include \
-D CMAKE_C_FLAGS=-I/usr/local/arm-none-linux-gnueabi/include \
-D BUILD_NEW_PYTHON_SUPPORT=0 \
-D WITH_OPENEXR=0 \
-D WITH_1394=0 \
-D WITH_GTK=0 \
-D HAVE_GTHREAD=0 \
-D BUILD_EXAMPLES=0 \
-D BUILD_TESTS=0 \
-D BUILD_SHARED_LIBS=0 \
-D X86=0 \
-D CMAKE_INSTALL_PREFIX=../install-2.4/ \
-D BUILD_ZLIB=1 \
-D BUILD_JPEG=1 \
-D BUILD_JASPER=1 \
-D BUILD_TIFF=0 \
```

```
-D WITH_TIFF=0 \
-D BUILD_PNG=1 \
-D ENABLE_SSE=0 \
-D ENABLE_SSE2=0 \
-D ENABLE_SSE3=0 \
-D ENABLE_SSSE3=0 \
-D ENABLE_SSE41=0 \
-D ENABLE_FAST_MATH=0 \
-D BUILD_PERF_TESTS=0 \
-D WITH_V4L=1 \
../OpenCV
```

Yukarıda ki komutta istenilen özellikler aktif ya da pasif olarak değiştirilebilir. Ayrıca çapraz derleme aracı sistem path'ine ekli ise, tam yolu göstermek yerine sadece çapraz derleme aracının ismi yazılabilir. -D BUILD_SHARED_LIBS=0" özelliği ile dinamik olarak derleme yapacağımız bildirilmektedir. Eğer statik derleme yapma isteniliyorsa bu özellik "1" olarak atanmalıdır. Özellikleri aktif ya da pasif yapmak için "0/1" yerine "ON/OFF" kelimeleri de kullanılabilir. "-D CMAKE_INSTALL_PREFIX=../install-2.4/ " parametresi, derlenen dosyaların nereye oluşturulacağını bildirmektedir. Bu dizin yolu da isteğe bağlı olarak değiştirilebilir. Bu komut gerekli ayarlamaları yapacak ve kuruluma hazır hale getirecektir.

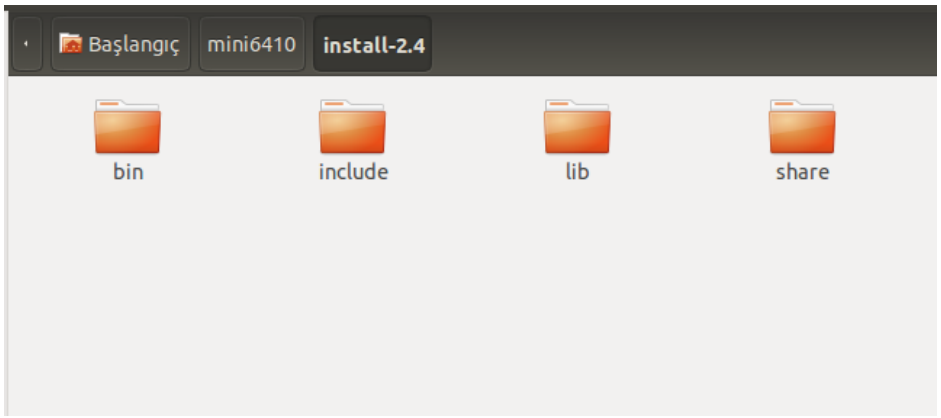
OpenCV gömülü sistemler için bazı kısıtlar getirmiştir. Örneğin; gömülü sistemlerde uygulama geliştirilirken, kamera çözünürlüğü değiştirilememektedir. Bu nedenle gömülü sistemlerde çalışmaya başlamadan önce kullanılacak kamera çözünürlüğü tespit edilmeli ve derleme yapmadan önce kamera çözünürlüğü ayarlanmalıdır. Derleme esnasında çözünürlüğü ayarlamak için "OpenCV/modules/highgui/src/cap_v4l.cpp" dosyasında değişiklik yapmak gerekmektedir. Bu dosya açılarak "static int icvSetVideoSize(CvCaptureCAM_V4L* capture, int w, int h) {}" fonksiyonunun tanımlandığı yer bulunur ve fonksiyonun en üstüne "h" parametresi için çözünürlüğün yükseklik, "w" parametresi için çözünürlüğün genişlik bilgisi verilmesi gerekmektedir. Bu tez çalışmasında kullanılan çözünürlük h = 180; w = 240 olarak kullanılmıştır. Böyle bir değişiklik

yapıldığında fonksiyona hangi çözünürlük değeri gelirse gelsin, kullanıcının istediği çözünürlük değeri çalışacaktır.

Bu ayarlama yapıldıktan sonra derleme işlemi yapılabilir. Linux sistemlerde derleme işlemi komut satırında tek tek yapılabileceği gibi "MakeFile" dosyaları oluşturulup, tüm komutlar bu dosyaya yazılıp ve terminalden sadece "make" komutu verilerek tek adımda da derleme işlemi yapılabilir. CMAKE komutu kullanıcı için "MakeFile" dosyasında gerekli düzenlemeleri, kendisine verilen özelliklere göre yapmaktadır. OpenCV dizinine terminal üzerinden gelinerek "make" komutu çalıştırılır. Bu komut çalıştırılmadan önce sistem yöneticisi izinlerine sahip olunduğundan emin olunmalıdır. "make" komutu sistemin performansına göre vakit alabilmektedir. Ek olarak "make" komutu derleme esnasında kullanılacak işlemci thread sayısını almaktadır. "make –j*" komutunda * yerine thread sayısı parametre olarak verilebilir. Thread sayısı her ne kadar arttıkça derleme süresi kısalsa da, fazla değerler verildiğinde tam tersine süreyi uzatabilmektedir. En ideal olanı sistemin çekirdek sayısını parametre olarak vermektir.

Make işlemi, dosyaların belirtilen dizine oluşturulması haricinde tüm işlemleri yapmaktadır. Kullanıcı make işleminde bir hata yapmış ise dosyalar oluşturulmadan önce değişiklik yapabilmesi için make ve make install olmak üzere iki aşamadan oluşmaktadır. Eğer herhangi bir hata ya da değişiklik yapılacaksa tekrar make install işlemi yapılmadan önce "make clean" komutu ile tüm derleme ayarları sıfırlanıp en başa döndürülmelidir. Aksi takdirde bir önceki ayarlar aynen kalacaktır. "MakeFile" dosyası ile derleme yapıldığı zaman, sistem oluşacak dosyaları, ayarları önceden yapılıp yapılmadığını kontrol eder. Zaman kaybını önlemek için eğer önceden oluşmuş dosya ya da ayarlar mevcut ise, o işlemleri atlar. Bu nedenle, eğer make işleminde değişiklik yapılacak ise önceki tüm ayar ve dosyaların silinmesi gerekmektedir. Tüm ayar ve dosyaları teker teker silmek hem zor hem de gözden kaçabileceği için "make clean" komutu kullanılmalıdır. Bu komut silinmesi gereken tüm ayar dosyalarını silmektedir. Dosya ve ayarları sıfırlamadan tekrar "make" komutu çalıştırılırsa ilk seferde dakikalar hatta saatler süren make işlemi, tüm dosya ve ayarlar mevcut olduğundan saniyeler sürdüğü gözlemlenecektir.

Make işlemi tamamlandığında, derlenen dosyaların belirtilen dizine oluşturulabilmesi için “make install” komutunun çalıştırılması gerekmektedir. Bu komut tamamlandığında CMAKE ayarlaması yapılırken belirtilen dizinin oluştuğu görülecektir. Bu dizin ARM işlemciler için OpenCV kütüphanelerini içermektedir (Şekil I.1). Qt programına harici kütüphane eklenirken, gömülü sistemler için, bu dizinde bulunan kütüphaneler eklenerek uygulamalar geliştirilebilir.



Şekil I.1. OpenCV derleme sonucu oluşan dizinler

EK J. ED-MK4 Sürücü devresi seri port haberleşme komutları

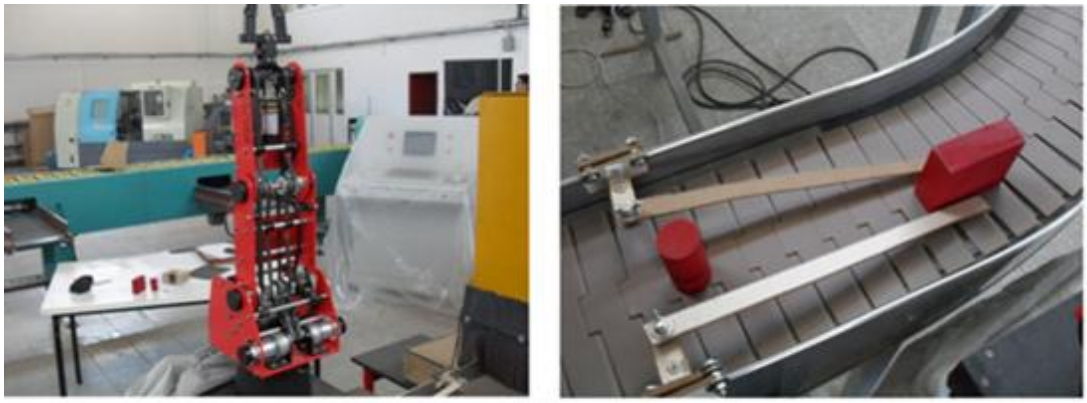
ED-MK4 sürücü devresi için seri port haberleşme komutlarına Çizelge J.1'den bakılabilir.

Çizelge J.1. ED-MK4 sürücü devresi seri port haberleşme komutları

Komut	Açıklama	Geri Dönüş	Parametre
SA	Hangi motor hareket ediyor bilgisi	Sayısal olarak dönüş yapar	
CG	Tutucuyu aktif ya da pasif yapma		0 = Pasif 1 = Aktif
GS	Tutucu durumunu geri döndürür	1 ise kapalı, 0 ise açık	
PA, m	Motor pozisyon bilgisini döndürür	Sayısal olarak dönüş yapar	m=A, B, C, D, E, F
GC	Tutucuyu kapat		
GO	Tutucuyu aç		
HH	Başlangıç pozisyonuna git		
MA	Tüm hareketi durdur		
MC	Tüm motorları aynı anda verilen konumlara hareket ettir		
PR, m, d	Son kaldığı konumunu referans alarak motorları verilen açı kadar hareket ettir		m=A, B, C, D, E, F d = açı bilgisi -32767<=d<=32767
PD, m, d	Başlangıç konumunu referans alarak motorları verilen açı kadar hareket ettir		m=A, B, C, D, E, F d = açı bilgisi -32767<=d<=32767

EK K. Uygulama resimleri

Bu bölümde, uygulamaya ait resimler bulunmaktadır. Şekil K.1-K.8 resimlerinde robot kolun ürün alma, bırakma ve robot kolun başlangıç, ürün alma ve ürün bırakma noktaları gösterilmiştir.

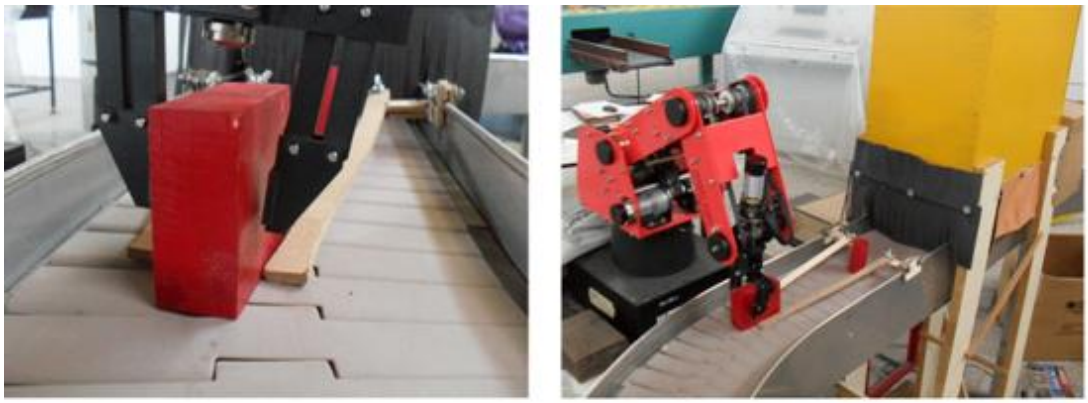


(a) (b)

Şekil K.1. Uygulama resimleri (a) Robot kol başlangıç konumu (b) Ürün hizalama aparatı

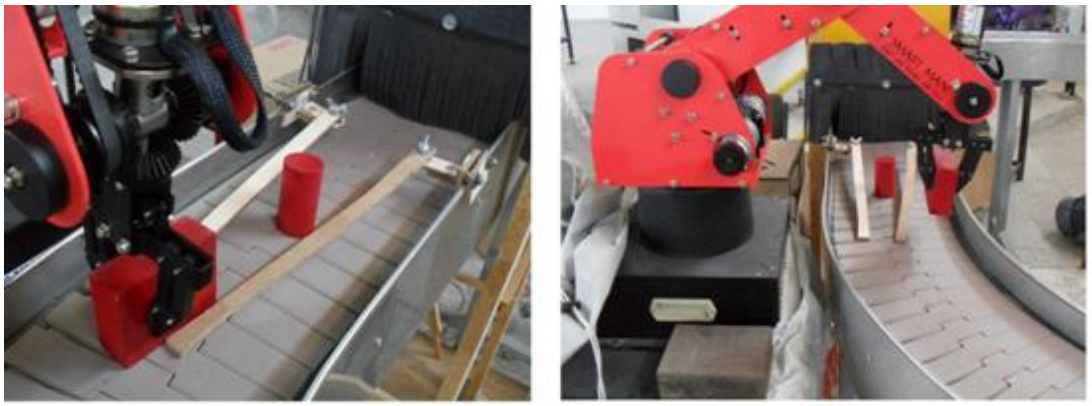
(a) (b)

Şekil K.2. Uygulama resimleri (a) Robot kol ve kamera kutusu (b) Ürün alma işlemi



(a) (b)

Şekil K.3. Uygulama resimleri (a) Ürün alma işlemi (b) Ürün alma işlemi karşıdan görünüş



(a) (b)

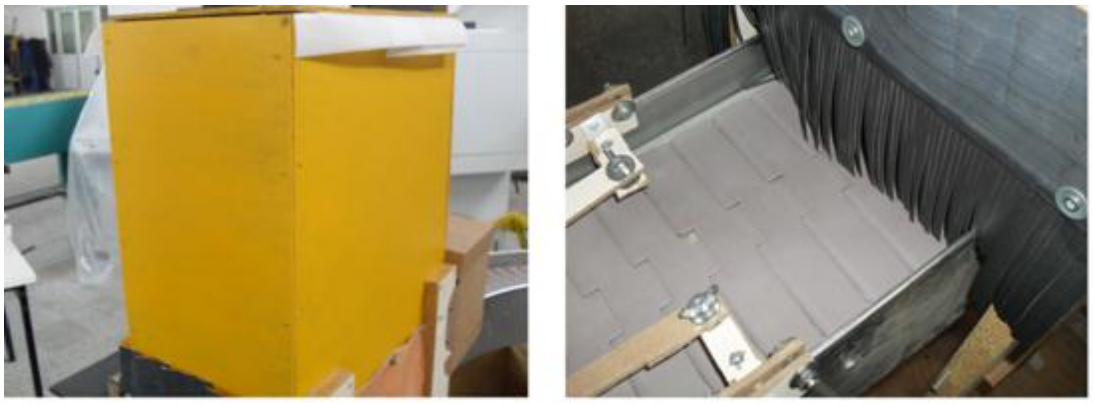
Şekil K.4. Uygulama resimleri (a) Robot kol tutma işlemi (b) Ürün banttan alma işlemi

(a) (b)

Şekil K.5. Uygulama resimleri (a) Robot kol ürün bırakma konumu (b) Robot kol ürün bırakma konumu yandan görünüş

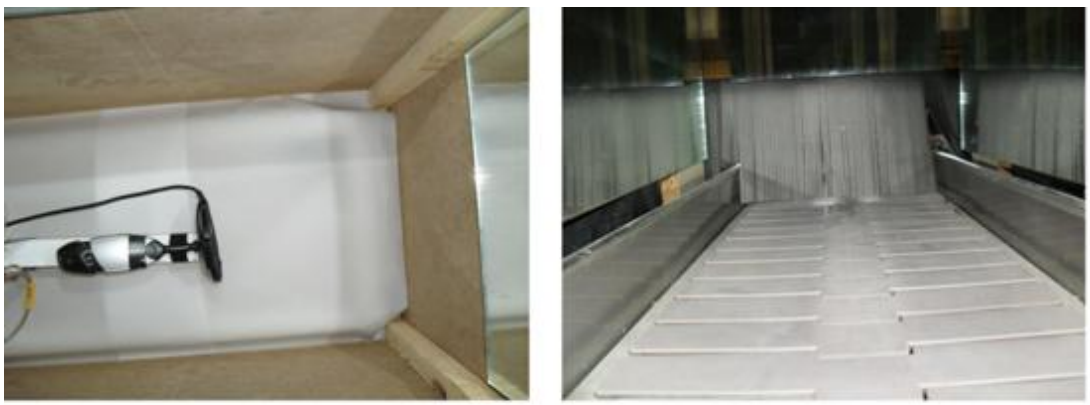
(a) (b)

Şekil K.6. Uygulama resimleri (a) Robot kol ürün bırakma işlemi (b) Robot kol başlangıç konumu



(a) (b)

Şekil K.7. Uygulama resimleri (a) Kamera görüntü alma kapalı kutusu (b) Kapalı kutu çıkışı



(a) (b)

Şekil K.8. Uygulama resimleri (a) Kapalı kutu kamera yerleşimi (b) Kapalı kutu yürüyen bant görüntüsü

ÖZGEÇMİŞ

Adı Soyadı : Fatih Ahmet ŞENEL

Doğum Yeri ve Yılı : Malatya, 1988

Medeni Hali : Evli

Yabancı Dili : İngilizce

E-posta : fatihsenel@sdu.edu.tr

Eğitim Durumu

Lise : Malatya Org. Eşref Bitlis Lisesi, 2005

Lisans : Pamukkale Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği

Yüksek Lisans : -

Mesleki Deneyim

SDÜ TEKNOKENT Targenik Ltd. Şti.	2010-2011
SDÜ Mühendislik Fakültesi Bilgisayar Mühendisliği	2011-……..(halen)

Yayınları

Şeker A.K., Çağlar M. F., Şenel F. A., Uz V. K., 2011. Lazer Mesafe Sensörü ile Yüzey Tarama Sistemi, Fırat Üniversitesi, Elektrik-Elektronik Bilgisayar Sempozyumu (FEEB 2011), Bildiriler Kitabı II, Elazığ, s. 56-60.

Şenel F.A., Tokat S., 2012. Görüntü İşleme Teknikleri Kullanılarak Bir Ortamın İnsan Yoğunluğunun Hesaplanması Elektrik - Elektronik ve Bilgisayar Mühendisliği Sempozyumu (ELECO '2012), Bursa, s. 613-617.