Android Platformunda OpenCV İle Görüntü İşleme

Research · January 2017

DOI: 10.13140/RG.2.2.11195.62245

CITATIONS
0

READS
5,762

1 author:

Çiğdem Çavdaroğlu
Isik University
14 PUBLICATIONS **10** CITATIONS

SEE PROFILE

Android Platformunda OpenCV İle Görüntü İşleme

Çiğdem Çavdaroğlu

12.05.2015

Özgeçmiş

- Yıldız Teknik Üniversitesi
  - 2003: Matematik Mühendisliği, Lisans
  - 2006: UA, Fotogrametri ve CBS, Yüksek Lisans
  - 2013: UA, Fotogrametri ve CBS, Doktora
- KoçSistem Ar-Ge, Analiz Tasarım Lideri
- Tübitak 1512 Bireysel Girişimcilik Destek Programı

Ajanda

- OpenCV & OpenCV4Android
- Geliştirme Ortamı ve Araçlar
  - Android SDK
  - Android NDK
  - OpenCV SDK
- Uygulama Geliştirme
  - Java ve OpenCV
  - C++ ve OpenCV
- Android tabanlı cihazlarda görüntü işleme
  - Görüntü İşleme ve OpenCV örnekleri

OpenCV ve OpenCV4Android

OpenCV

- OpenCV - Open Source Computer Vision
- Gerçek zamanlı bilgisayar ile görme kütüphanesi.
- Çıkış yılı: 1999
- C ve C++ programlama dilleri ile geliştirilmiş
- BSD (Berkeley Software Distribution) lisansına sahip

OpenCV Desteği Bulunan İşletim Sistemleri

- Linux
- Windows OS x
- Android
- iOS
- BlackBerry 10

OpenCV Desteği Bulunan Programlama Dilleri

- C/C++
- Java
- NET
- PHP
- MATLAB
- OCTAVE
- Python
- Ruby
- Delphi

OpenCV Kütüphanesi

- core: Temel modül
  - Temel veri yapılarının tanımları
  - Diğer modüller tarafından da kullanılan temel fonksiyonlar
- imgproc : Görüntü işleme modülü
  - Doğrusal ve doğrusal olmayan görüntü filtreleme
  - Geometrik görüntü dönüşümleri (yeniden boyutlandırma, afin dönüşümü, perspektif dönüşüm)
  - Renk uzayı dönüşümleri
  - Histogram fonksiyonları vb
- Video: Video analiz modülü
  - Hareket analizi (Motion estimation)
  - Arka plan çıkarımı işlemleri (Background subtraction)
  - Nesne izleme algoritmaları (Object tracking algoritmaları)
- calib3d
  - Çoklu görüş geometri algoritmaları (Multiple-view geometry algorithms)
  - Tek resimli ve stereo kamera kalibrasyon algoritmaları
  - Nesne poz değerlendirme algoritmaları (Object pose estimation)
  - Stereo eşleştirme algoritmaları (Stereo correspondence)
  - 3B geri çatım (3D reconstruction)
- features2d
  - Detay nokta yakalayıcılar, tanımlayıcılar ve eşleştiriciler (salient feature detectors, descriptors, and descriptor matchers)
- objdetect
  - Önceden tanımlanmış nesnelerin yakalanması
- highgui
  - Video verisi işleme
- Gpu
  - Çeşitli OpenCV modüllerinde yer alan fonksiyonların hızlandırılmış versiyonları

OpenCV4Android

- 2010 yılı başlarında OpenCV 2.2 versiyonu kullanılırken Android platformu kısıtlı olarak desteklenmeye başladı.
- OpenCV 2.3.1 ile «OpenCV for Android» beta olarak sürüldü. Bu başlangıç versiyonu OpenCV Java API'si ve doğal (native) kamera desteği içeriyordu.
- İlk resmi sürüm, OpenCV 2.4 versiyonu ile Nisan 2012 yılında gerçekleşti.

OpenCV4Android'e Başlarken

- İki temel tip başlangıç seviyesi:
  - OpenCV deneyimli, Android'e yeni başlayan
    - Mobil işletim sistemleri & masaüstü işletim sistemleri farklılıklar
    - Android'i tanımak
      - Introduction into Android Development (http://docs.opencv.org/doc/tutorials/introduction/android_binary_package/android_dev_intro.html)
      - Android Lifecycle For Application Developers: Guidelines and Tips (https://developer.nvidia.com/sites/default/files/akamai/mobile/docs/android_lifecycle_app_note.pdf)
    - C++'da mevcut fonksiyonlarımızı kullanmak
    - Android ve diğer işletim sistemleri arasındaki performans farklılıkları
    - Android ve normal OpenCV sürüm farklılıkları

OpenCV4Android'e Başlarken

- İki temel tip başlangıç seviyesi:
  - Android deneyimli, OpenCV'ye yeni başlayan
    - OpenCV'yi tanımak:
      - OpenCV dokümantasyonu (http://docs.opencv.org/)
      - Dokümanlar (http://docs.opencv.org/doc/tutorials/tutorials.html)
      - Forumlar (http://answers.opencv.org/questions/)
      - Wiki (http://code.opencv.org/projects/opencv/wiki)

- Yol göstericiler:
  - http://opencv.org/platforms/android.html
    - OpenCV4Android dokümantasyonu
    - OpenCV ile ilgili temel bilgiler
    - Harici kütüphaneler ve uygulamalar

Geliştirme Ortamı ve Araçlar
Android SDK & NDK
OpenCV SDK

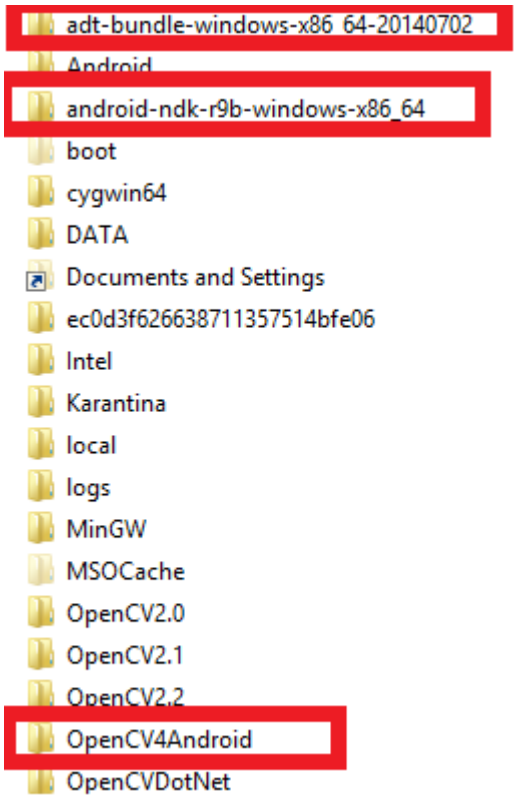
Geliştirme Ortamı Araçları ve Bileşenleri

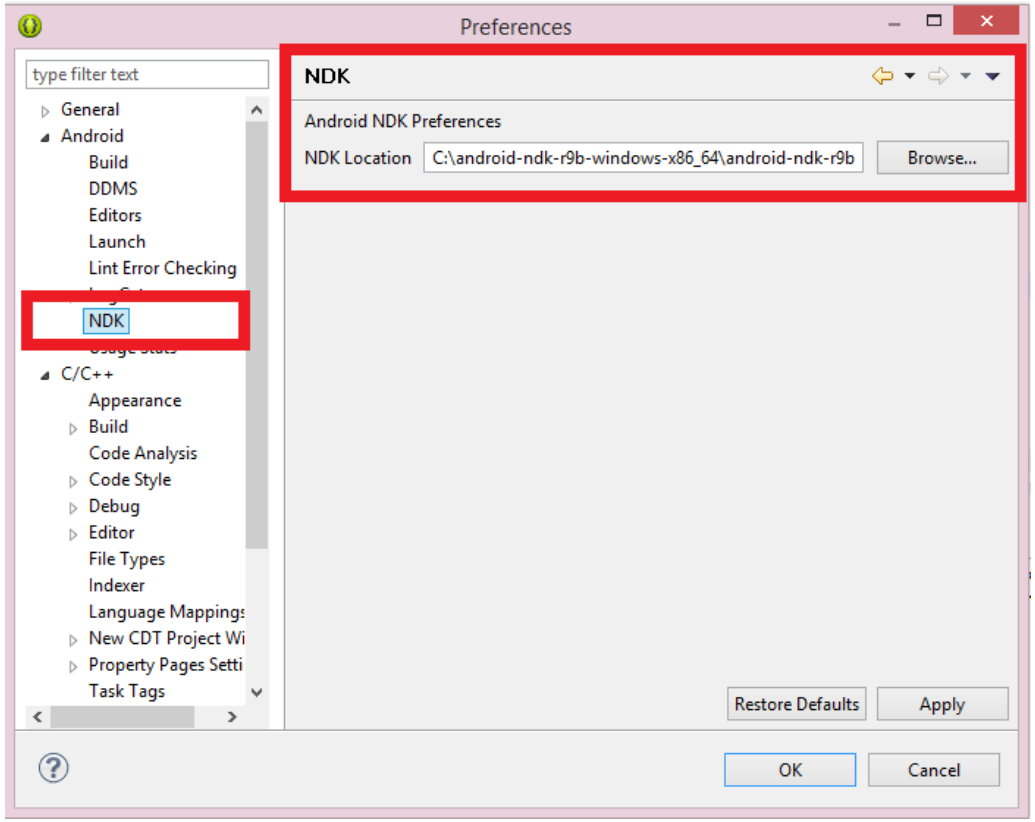
- JDK
- SDK
- NDK
- Derleyici IDE
- OpenCV4Android SDK
  - OpenCV Manager APK
  - Örnek Uygulamalar

Kurulumlar

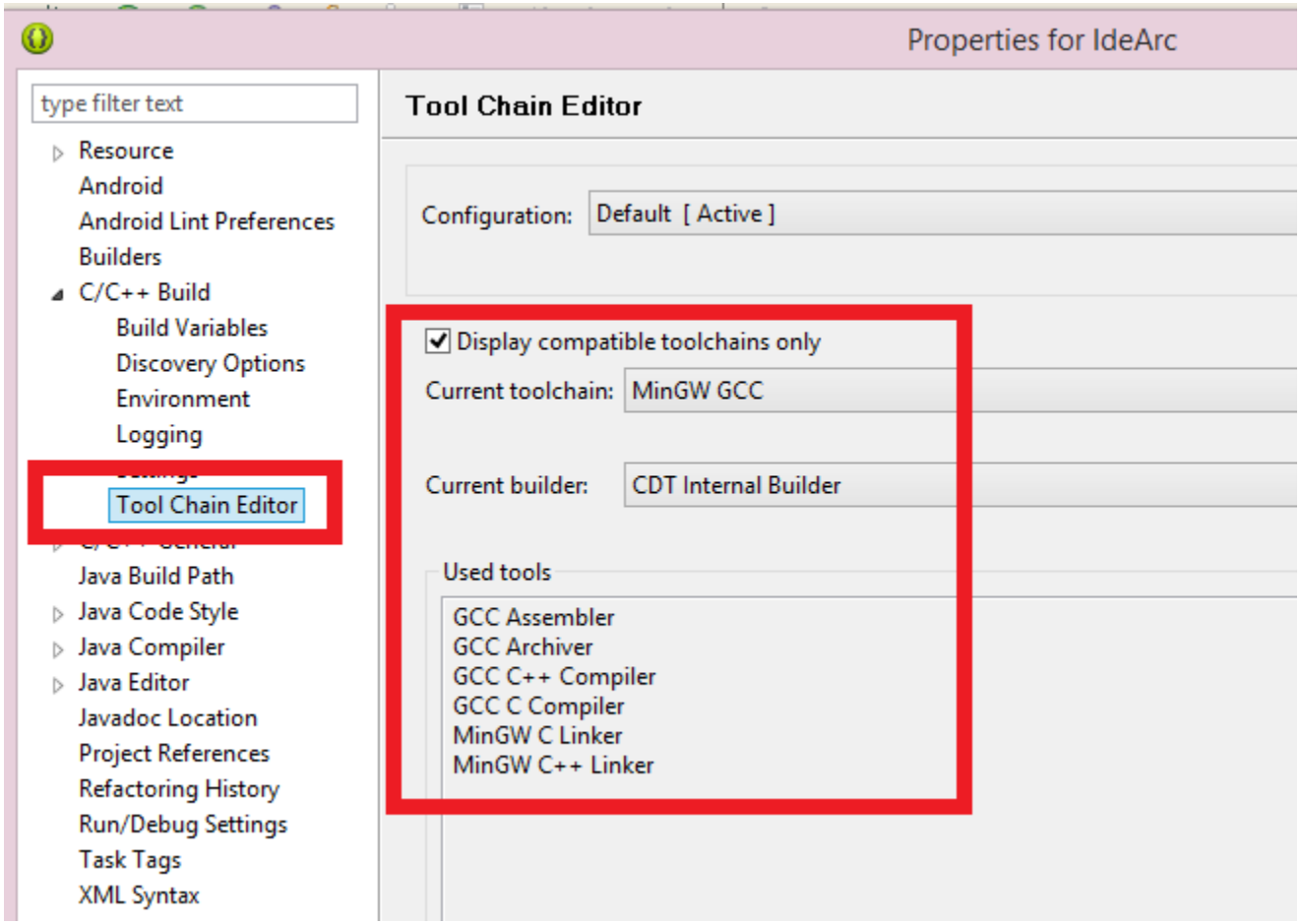
- Java SDK
  - http://www.oracle.com
- Android SDK
  - http://developer.android.com/sdk/index.html
- Android NDK
  - http://developer.android.com/tools/sdk/ndk/index.html
- Eclipse IDE
  - http://www.eclipse.org/downloads/
- Android Development Tools (ADT) plugin for Eclipse
  - http://developer.android.com/sdk/installing/installing-adt.html
- C/C++ Development Tooling (CDT) for Eclipse
- OpenCV4Android kütüphanesi:
  - SourceForge sitesinden indirilebilir. (OpenCV-x.x.x-android-sdk.zip)

Ayarlar


adt-bundle-windows-x86_64-20140702
Android
android-ndk-r9b-windows-x86_64
boot
cygwin64
DATA
Documents and Settings
ec0d3f626638711357514bfe06
Intel
Karantina
local
logs
MinGW
MSOCache
OpenCV2.0
OpenCV2.1
OpenCV2.2
OpenCV4Android
OpenCVDotNet


Preferences
type filter text
General
Android
Build
DDMS
Editors
Launch
Lint Error Checking
NDK
C/C++
Appearance
Build
Code Analysis
Code Style
Debug
Editor
File Types
Indexer
Language Mappings
New CDT Project Wi
Property Pages Setti
Task Tags
NDK
Android NDK Preferences
NDK Location
C:\android-ndk-r9b-windows-x86_64\android-ndk-r9b
Browse...
Restore Defaults
Apply
OK
Cancel

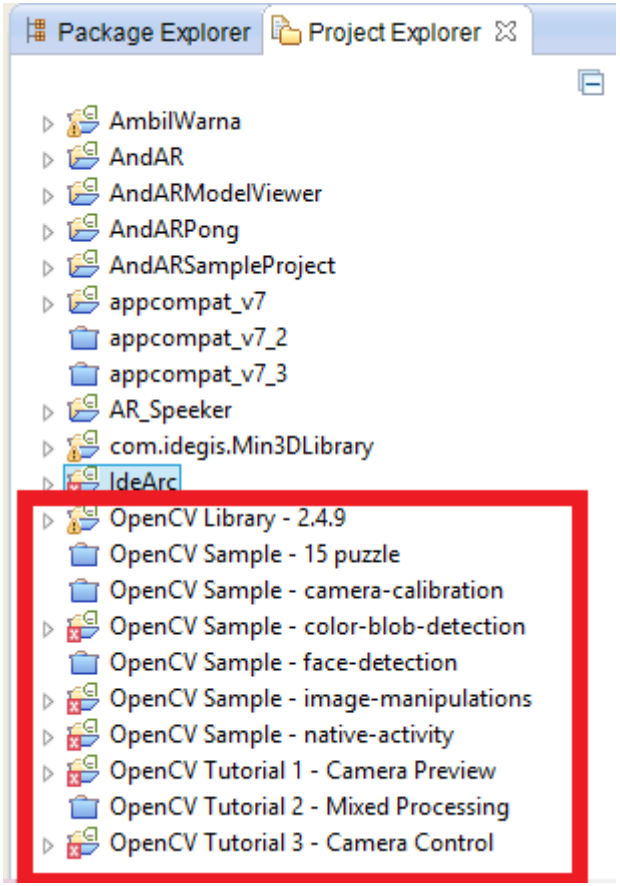
Ayarlar


Properties for IdeArc
type filter text
Resource
Android
Android Lint Preferences
Builders
C/C++ Build
Build Variables
Discovery Options
Environment
Logging
Tool Chain Editor
Java Build Path
Java Code Style
Java Compiler
Java Editor
Javadoc Location
Project References
Refactoring History
Run/Debug Settings
Task Tags
XML Syntax
Tool Chain Editor
Configuration: Default [Active]
Display compatible toolchains only
Current toolchain: MinGW GCC
Current builder: CDT Internal Builder
Used tools
GCC Assembler
GCC Archiver
GCC C++ Compiler
GCC C Compiler
MinGW C Linker
MinGW C++ Linker

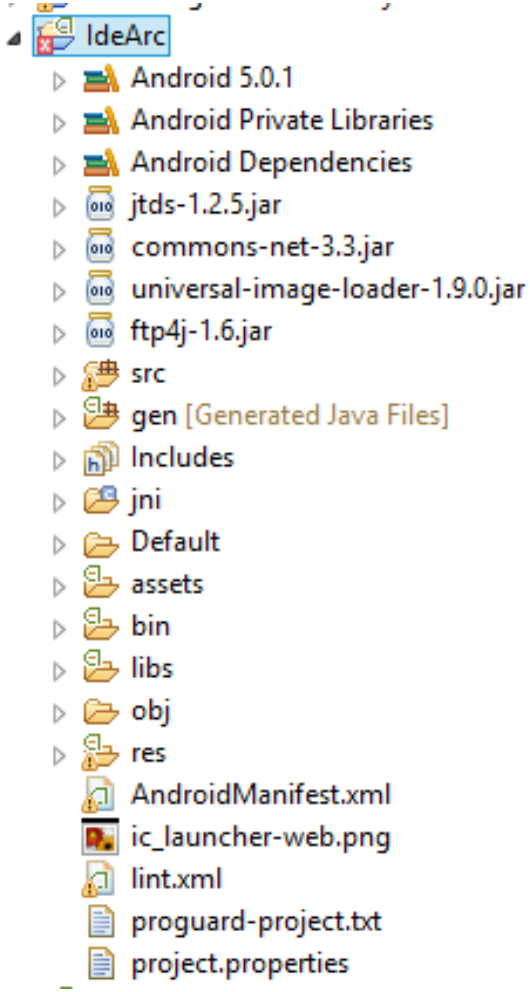
Ayarlar

- OpenCV4Android için ayarlar:
  - C:/ de yeni bir çalışma alanına kütüphane içeriği kopyalanır: «C:/OpenCVAndroid/»
  - İndirilen .zip dosyası bu dizinde açılır.
  - Kütüphane geliştirme ortamına eklenir: Eclipse için Package Explorer'da «import» komutu ile
  - Örnekler geliştirme ortamına eklenir.

- (http://docs.opencv.org/doc/tutorials/introduction/android_binary_package/O4A_SDK.html)



Android Proje Yapısı



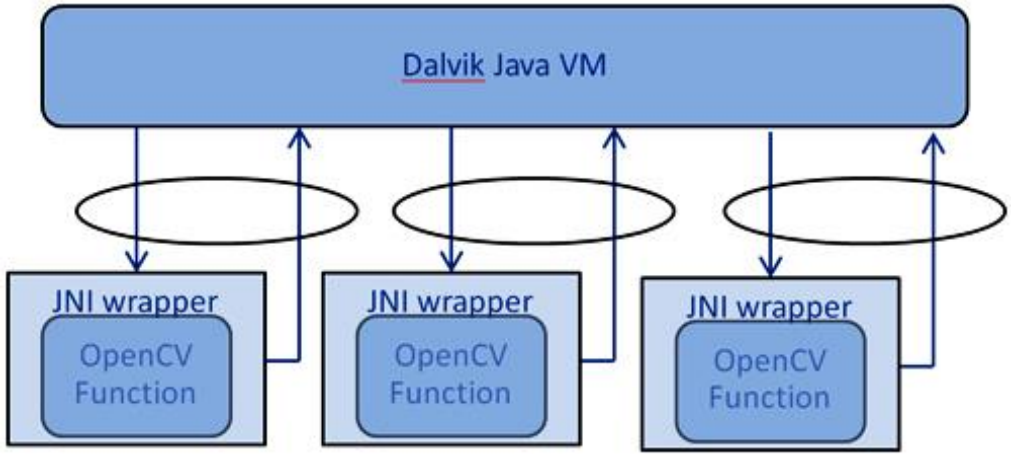
- Jni:
  - C/C++ kodu yazıldıysa oluşacaktır.
  - C/C++ uygulama kaynak kodlarını içerir.
  - Makefile sözdiziminde yazılmış Android.mk ve Application.mk dosyalarını içerir.
- Libs:
  - Derlenmiş kütüphaneleri içerir.
- Res:
  - Uygulama kaynak dosyalarını (resimler ve UI tasarımlarını tanımlayan XML dosyaları) içerir.
- Src:
  - Uygulamanın Java kodlarını içerir.
- AndroidManifest.xml:
  - Android sistemine ilişkin uygulama bilgilerini içerir.
- default.properties:
  - Hedeflenen Android platformu ve diğer build detay bilgilerini tanımlar.

Uygulama Geliştirme
Java ve OpenCV
C++ ve OpenCV

OpenCV4Android Geliştirme Biçimleri

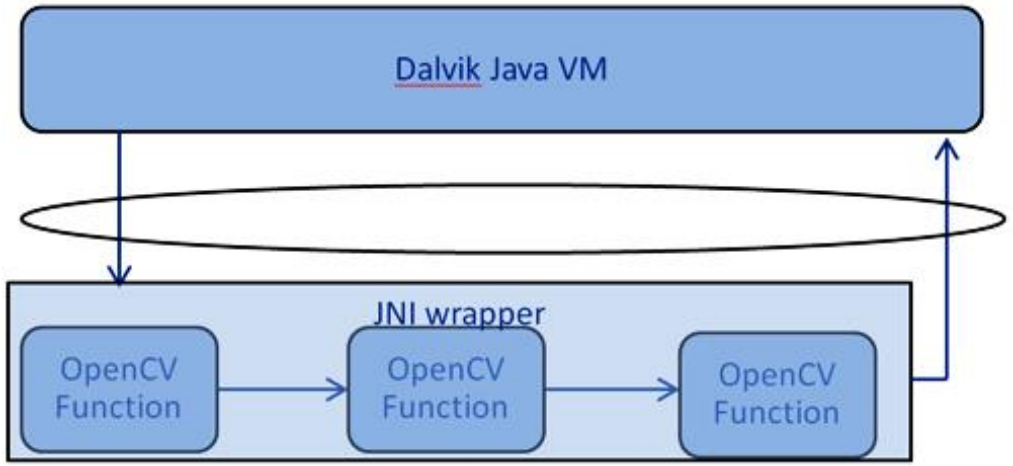
- Üç farklı geliştirme biçimi:
  - Temel düzey: OpenCV Java API'sini kullanmak.
    - OpenCV'de yer alan fonksiyonların çoğunluğu Java API'sinde yer almaktadır.
    - Geliştirmeler sırasında kullanılan fonksiyonların Java API'si içerisinde yer alıp almadığı mutlaka kontrol edilmeli ve geliştirme düzeni ona göre kurulmalıdır.
    - http://docs.opencv.org/java/ sitesinden fonksiyon listesi kontrol edilebilir.
    - OpenCV Java API + Android SDK yeterli
  - İleri düzey: Native ortamı kullanmak.
    - Android NDK kullanımı zorunludur.
    - Performans olarak daha yüksek uygulamalar geliştirmemizi sağlar.
    - Geliştirmesi ve geliştirme ortamının kurulumu daha zordur.
    - OpenCV işlemleri C++ ile yazılır. OpenCV fonksiyonları doğrudan çağrılır. Tüm OpenCV çağrıları tek bir sınıf içerisinde toplanabilir.
    - Her görüntü için bir kere çağrı yapılması yeterlidir.
    - JNI (Java Native Interface) çağrıları görüntü bazında azaltılır (iki çağrı).
    - OpenCV native interface + Android NDK gerekli.
  - Uzman düzey:
    - OpenCV kodları gerekli

OpenCV4Android Geliştirme Biçimleri



Java API ile

C++ ile



Hangi Geliştirme Biçimini Seçmeliyim?

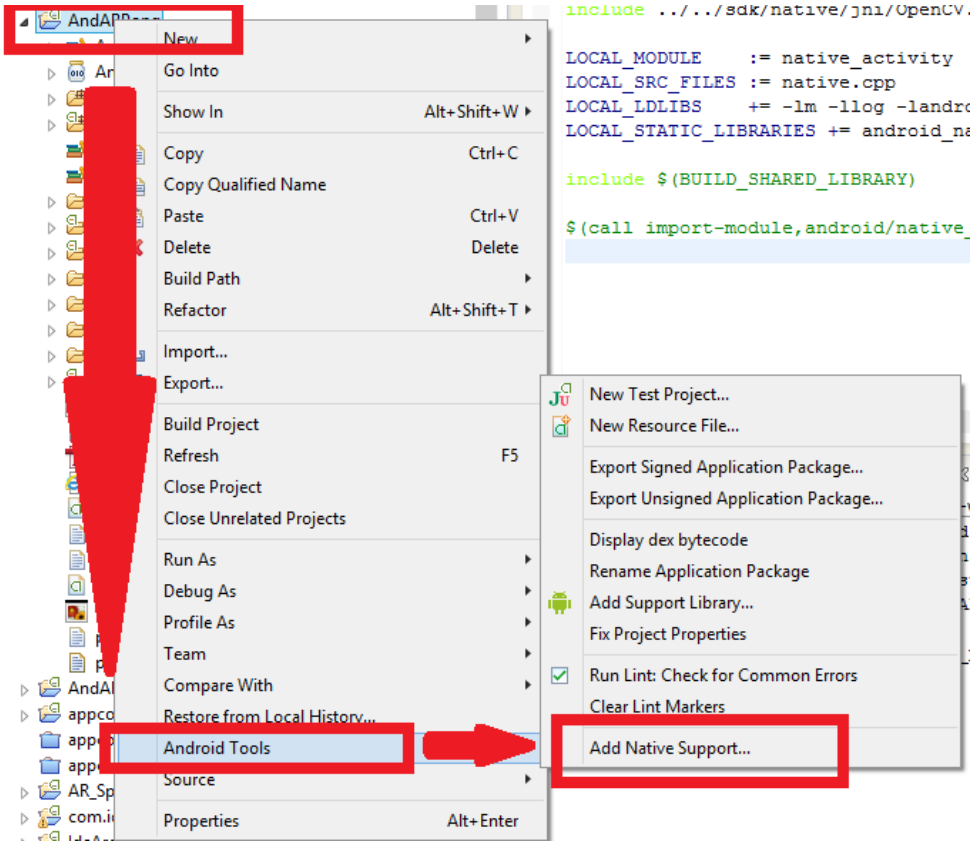
- Temel Düzey:
  - Java API yeterli mi?
  - Geliştirme yapmaya başlamak için OpenCV kütüphanesine referans vermek yeterlidir.
  - Geliştirme sonrasında JNI çağrılarının performansının uygulama kullanımına olan etkisi mutlaka kontrol edilmelidir.

- İleri Düzey:
  - Java API'sinin yeterli olmadığı durumda bu yol izlenmelidir.
  - Mevcut C/C++ kütüphane ve fonksiyonlarımızı kullanmak istiyorsak bu yol izlenmelidir.

- Uzman Düzey:
  - Her iki seçenekte de sunulan fonksiyonların yetersiz kaldığı noktada bu yol seçilmelidir.
    - Kullanmak istenilen OpenCV fonksiyonunda bir hata olabilir.
    - Uygulamaya özel olarak bir fonksiyonun farklı biçimde çalışmasını isteyebiliriz. Bu durumda OpenCV tarafındaki kod üzerinde değişiklik yapılmalıdır.
    - Yeni bir OpenCV fonksiyonu yazılması gerekli olabilir.
    - Mevcut ve düzgün çalışan bir fonksiyonda performans iyileştirmesi yapılması istenebilir.

Mevcut Projelere Doğal Dil Desteği Eklemek

- Proje özelliklerinden «Android Tools» seçeneği seçilir ve «Add Native Support» komutu verilir.
- Bu komutun verilmesinin ardından proje dizinine «jni» isimli dizin otomatik olarak oluşturulur.
- Bu dizin altında C++ dili ile geliştirilen kodlar ve konfigürasyon içeren mk uzantılı dosya bulunmaktadır.



Android Tabanlı Cihazlarda Görüntü İşleme OpenCV Örnekleri

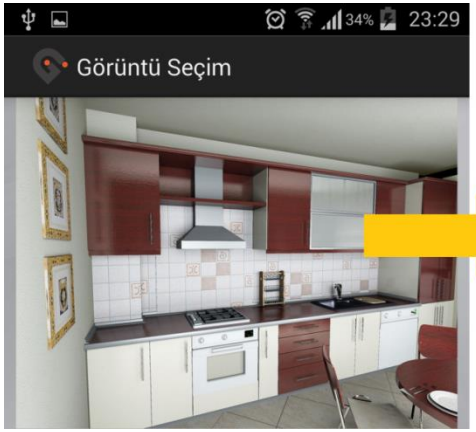
Java Kullanarak Görüntü İşleme

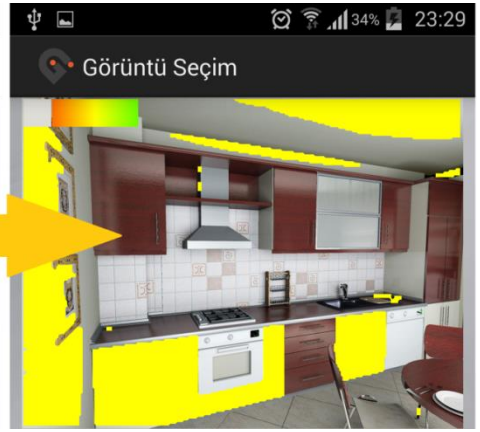

```
public  Bitmap applyBrightnessEffect(Bitmap src, int value)
{
    //Image size
    int width = src.getWidth();
    int height = src.getHeight();
    //Create output bitmap
    Bitmap bmOut = Bitmap.createBitmap(width, height,
    //Color information
    int A, R, G, B;
    int pixel;

    //Scan through all pixels
    for (int x = 0; x < width; ++x)
    {
        for (int y = 0; y < height; ++y)
        {
            //Get pixel color
            pixel = src.getPixel(x, y);
            A = Color.alpha(pixel);
            R = Color.red(pixel);
            G = Color.green(pixel);
            B = Color.blue(pixel);
```

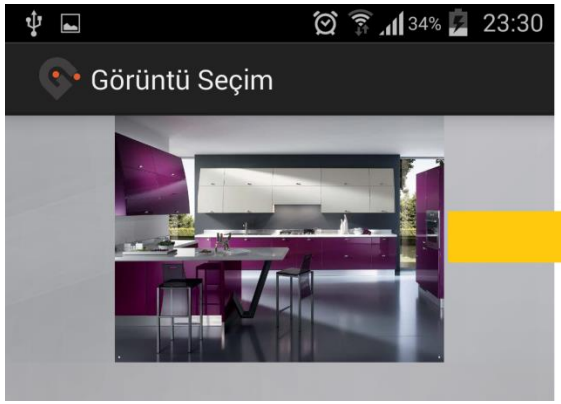
```
            //Increase/decrease each channel
            R += value;
            if(R > 255) { R = 255; }
            else if(R < 0) { R = 0; }

            G += value;
            if(G > 255) { G = 255; }
            else if(G < 0) { G = 0; }

            B += value;
            if(B > 255) { B = 255; }
            else if(B < 0) { B = 0; }

            //Apply new pixel color to output bitmap
            bmOut.setPixel(x, y, Color.argb(A, R, G, B));
        }
    }
```

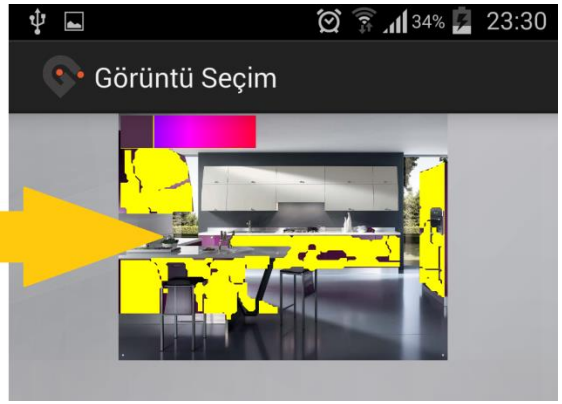
Renk Yakalama, Alan Belirleme ve Renk Değiştirme

- Bir görüntü üzerinde kullanıcının seçtiği alana ilişkin renk bilgisi alınır.
- Renge göre görüntüyü sınıflandıracak bir sınıf geliştirilir.









OpenCV
Java

Renk Yakalama, Alan Belirleme ve Renk Değiştirme

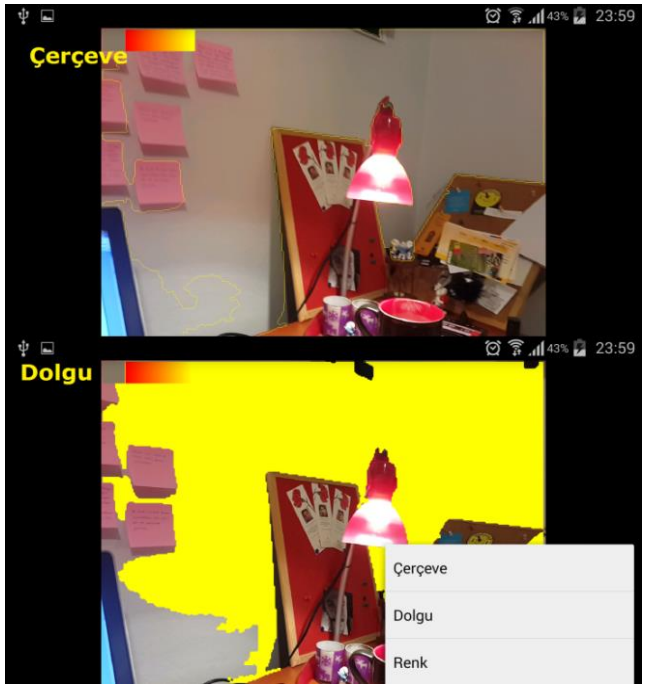
```
public boolean onTouch(View v, MotionEvent event)

    int columns = _rgba.cols();
    int rows = _rgba.rows();

    int xOffset = (_bitmap.getWidth() - columns)
    int yOffset = (_bitmap.getHeight() - rows) /

    //Touched coordinates on image
    int x = (int)event.getX() - xOffset;
    int y = (int)event.getY() - yOffset;

    if ((x < 0) || (y < 0) || (x > columns) || (y
        return false;

    Rect touchedRectangle = new Rect();

    touchedRectangle.x = (x>4) ? x-4 : 0;
    touchedRectangle.y = (y>4) ? y-4 : 0;

    touchedRectangle.width = (x+4 < columns) ? x
    touchedRectangle.height = (y+4 < rows) ? y +
```

```
Mat touchedRegionRgba = _rgba.submat(touchedRectangle);

Mat touchedRegionHsv = new Mat();
Imgproc.cvtColor(touchedRegionRgba, touchedRegionHsv, I

//Calculate average color of touched region
_blobColorHsv = Core.sumElems(touchedRegionHsv);
int pointCount = touchedRectangle.width * touchedRectan
for (int i = 0; i < _blobColorHsv.val.length; i++)
    _blobColorHsv.val[i] /= pointCount;

_blobColorRgba = converScalarHsv2Rgba(_blobColorHsv);
_detector.setHsvColor(_blobColorHsv);

Imgproc.resize(_detector.getSpectrum(), _spectrum, SPEC

_isColorSelected = true;

touchedRegionRgba.release();
touchedRegionHsv.release();

DrawImageContours();

return true;
}
```

Renk Yakalama, Alan Belirleme ve Renk Değiştirme

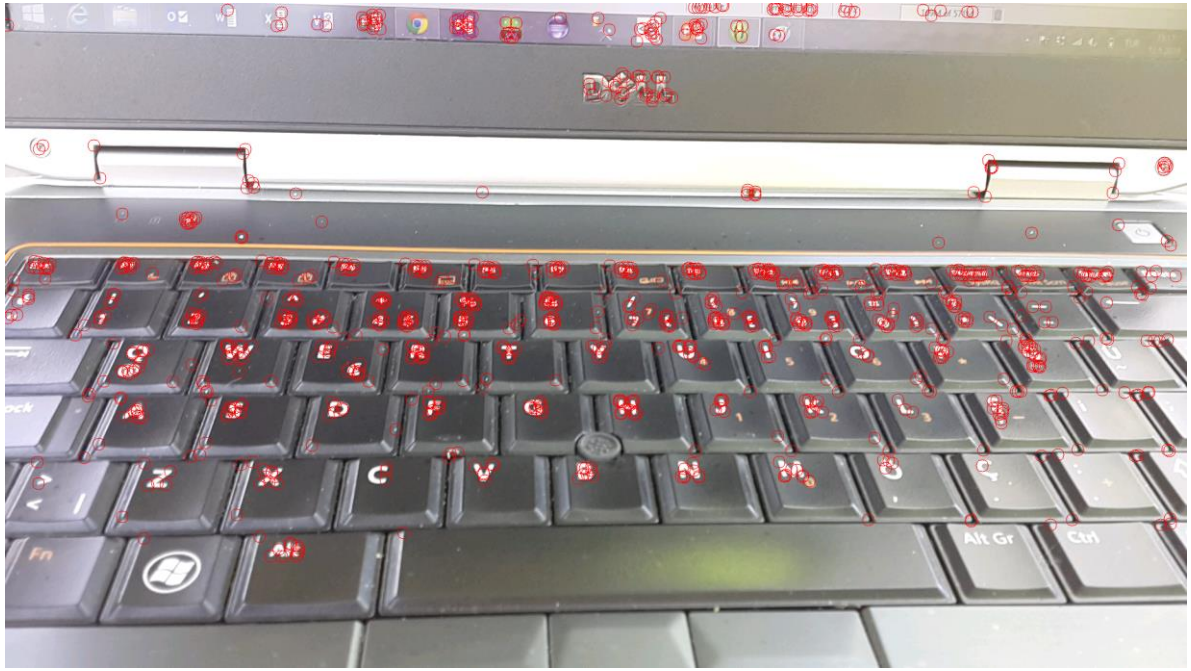
```
public void process(Mat rgbaImage) {
    Imgproc.pyrDown(rgbaImage, pyramidDownMat);
    Imgproc.pyrDown(pyramidDownMat, pyramidDownMat);

    Imgproc.cvtColor(pyramidDownMat, hsvMat, Imgproc.COLOR_RG

    Core.inRange(hsvMat, lowerBound, upperBound, mask);
    Imgproc.dilate(mask, dilatedMask, new Mat());

    List<MatOfPoint> contours = new ArrayList<MatOfPoint>();

    //Imgproc.findContours(dilatedMask, contours, hierarchy,
    Imgproc.findContours(dilatedMask, contours, hierarchy, Im

    //Find max contour area
    double maxArea = 0;
    Iterator<MatOfPoint> each = contours.iterator();
    while (each.hasNext()) {
        MatOfPoint wrapper = each.next();
        double area = Imgproc.contourArea(wrapper);
        if (area > maxArea)
            maxArea = area;
    }
```

- OpenCV findContours fonksiyonu ile belirtilen renge göre görüntü içerisindeki alanlar belirlenir.



Detay Nokta ve Kenar Yakalama

- Fast detay nokta yakalama operatörü ile canlı görüntü üzerinde detay noktaların yakalanması
- Canny kenar yakalama operatörü ile canlı görüntü üzerinde kenar bileşenlerinin yakalanması



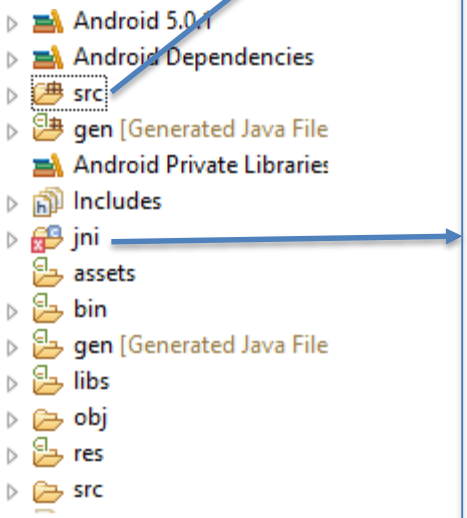
OpenCV

C++

Detay Nokta ve Kenar Yakalama

```
case VIEW_MODE_FEATURES:
    // input frame has RGBA format
    mRgba = inputFrame.rgba();
    mGray = inputFrame.gray();
    FindFeatures(mGray.getNativeObjAddr(), mRgba.getNativeObjAddr());
    break;
}
```

Detay Nokta ve Kenar Yakalama

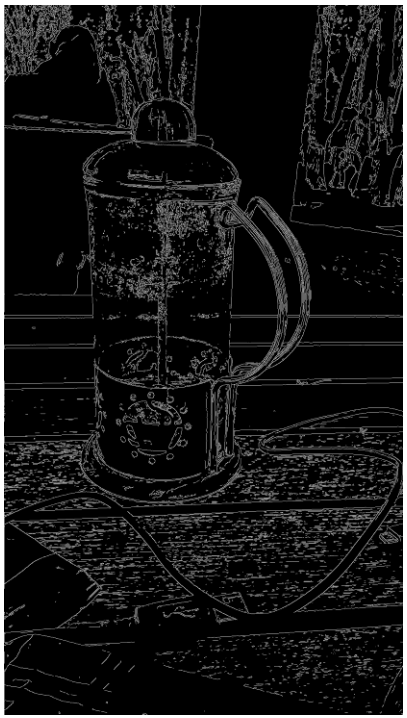


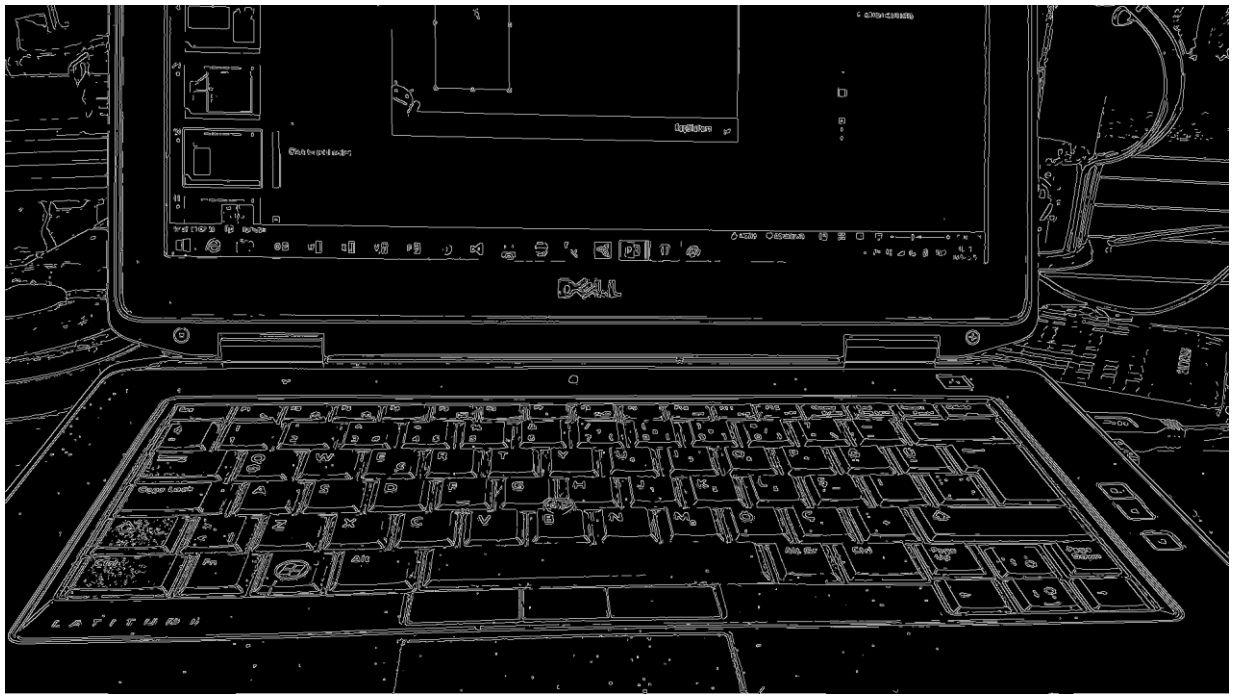

```
public native void FindFeatures(long matAddrGr, long matAddrRgba);
```

```
#include <jni.h>
#include <opencv2/core/core.hpp>
#include <opencv2/imgproc/imgproc.hpp>
#include <opencv2/features2d/features2d.hpp>
#include <vector>

using namespace std;
using namespace cv;

extern "C" {
JNIEXPORT void JNICALL Java_org_opencv_samples_tutorial2_Tutori

JNIEXPORT void JNICALL Java_org_opencv_samples_tutorial2_Tutori
{
    Mat& mGr  = *(Mat*)addrGray;
    Mat& mRgb = *(Mat*)addrRgba;
    vector<KeyPoint> v;

    FastFeatureDetector detector(50);
    detector.detect(mGr, v);
    for( unsigned int i = 0; i < v.size(); i++ )
    {
        const KeyPoint& kp = v[i];
        circle(mRgb, Point(kp.pt.x, kp.pt.y), 10, Scalar(255,0,
    }
}
}
```

Detay Nokta ve Kenar Yakalama

```
case VIEW_MODE_CANNY:
    // input frame has gray scale format
    mRgba = inputFrame.rgba();
    Imgproc.Canny(inputFrame.gray(), mIntermediateMat, 80, 100);
    Imgproc.cvtColor(mIntermediateMat, mRgba, Imgproc.COLOR_GRAY2RGBA, 4);
    break;
```





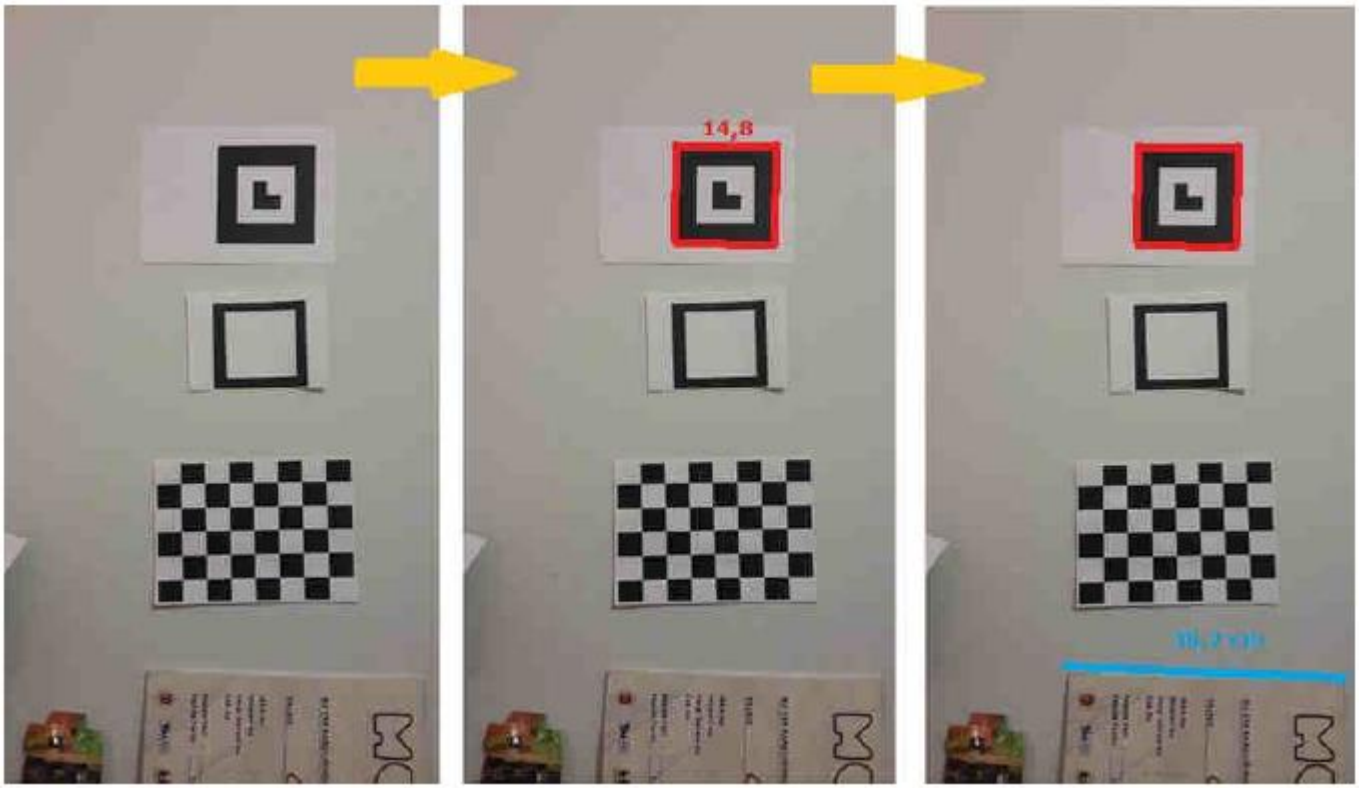
Detay Nokta Yakalama ve Eşleştirme

- Harris operatörünün kullanımı ile detay nokta yakalama ve eşleştirme
- Yakalanan detay noktaların iyileştirilmesi (Ransac algoritmasının uygulanması)
- Elde edilen çakışık noktaların kullanımı ile iki görüntünün birleştirilmesi

OpenCV
Java

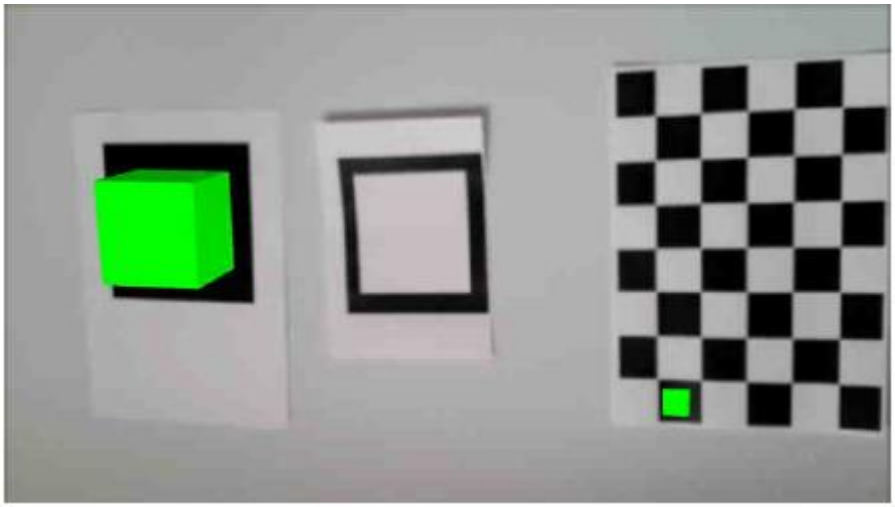
Diğer Örnekler

- Artırılmış Gerçeklik – İşaretleyici (marker) yakalama ve ölçülendirme



Diğer Örnekler

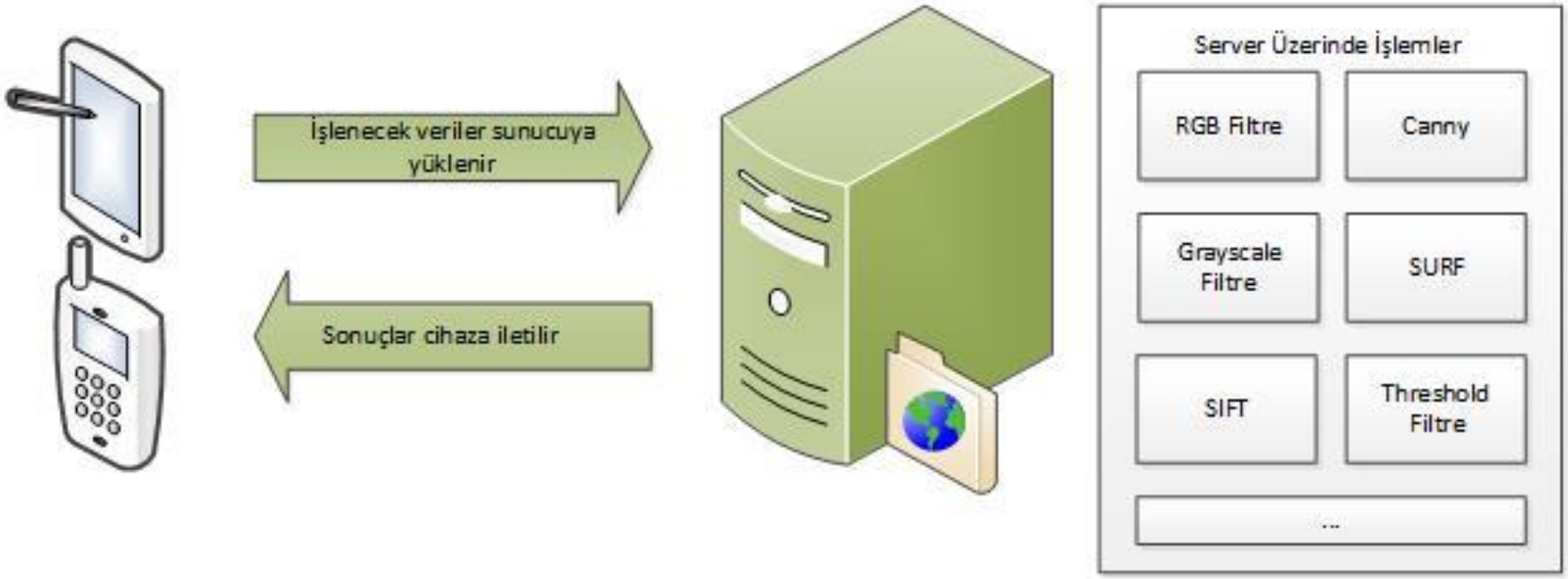
- Artırılmış Gerçeklik – Model Görüntüleme



Diğer Örnekler

- Artırılmış Gerçeklik – Model Görüntüleme

Server Desteği



TEŞEKKÜRLER

Çiğdem Çavdaroğlu