LAMPIRAN

JAVA

Cover.java

```java
package com.example.leafapp;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;

public class Cover extends Activity {


    @Override
    protected void onCreate(Bundle Cover) {
        super.onCreate(Cover);
        setContentView(R.layout.leaf);
        Thread timer = new Thread(){
            public void run(){
                try {
                    sleep(1000);
                }
                catch (InterruptedException e) {
                    e.printStackTrace();
                }
                finally {
                    Intent openMainActivity = new Intent("com.example.leafapp.MAINACTIVITY");
                    startActivity(openMainActivity);
                }
            }
        };
        timer.start();

    }

    @Override
    protected void onPause() {
        super.onPause();
        finish();
    }
}

```

MainActivity.java

```java
package com.example.leafapp;

import android.app.Activity;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.net.Uri;
import android.os.Bundle;
import android.provider.MediaStore;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;

import org.opencv.android.OpenCVLoader;
import org.opencv.android.Utils;
import org.opencv.core.Core;
import org.opencv.core.CvType;
import org.opencv.core.Mat;
import org.opencv.imgproc.Imgproc;

import java.io.FileNotFoundException;


public class MainActivity extends Activity {
    Button bCapt, bPickCal,bHelp;
    ImageView iv;
    Uri source;
    Bitmap bitmap, bitmapMaster;
    TextView Result;

    private static final int LOAD_IMAGE = 100;

    private static final String TAG = "MainActivity";

    static {
        if (!OpenCVLoader.initDebug()) {
            Log.d(TAG, "OpenCV not loaded");
        } else {
```

```
            Log.d(TAG, "OpenCV loaded");
        }
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        bCapt = (Button) findViewById(R.id.bCapture);
        bPickCal = (Button) findViewById(R.id.bPick);
        bHelp = (Button) findViewById(R.id.bHelp);
        iv = (ImageView) findViewById(R.id.imageView);
        Result = (TextView) findViewById(R.id.source);
        bitmap = BitmapFactory.decodeResource(this.getResources(), R.id.imageView);


        bCapt.setOnClickListener( new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(MainActivity.this, Application.class);
                startActivity(intent);
            }
        });
        bPickCal.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(Intent.ACTION_PICK, MediaStore.Images.Media.EXTERNAL_CONTENT_URI);
                startActivityForResult(intent, LOAD_IMAGE);
            }
        });
        bHelp.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(MainActivity.this, Help.class);
                startActivity(intent);
            }
        });

    }
```

```

    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);

        if (resultCode == RESULT_OK) {
            switch (requestCode) {
                case LOAD_IMAGE:
                    source = data.getData();

                    try {
                        bitmapMaster = BitmapFactory.decodeStream(getContentResolver().openInputStream(
                                source));
                        LoadBitmap(bitmapMaster);

                    } catch (FileNotFoundException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                    }

                    break;
            }
        }
    }

    private void LoadBitmap(Bitmap bitmap) {
        Mat mat = new Mat(bitmap.getWidth(), bitmap.getHeight(), CvType.CV_8UC1);
        // Convert
        Utils.bitmapToMat(bitmap, mat);

        Mat gray = new Mat(bitmap.getWidth(), bitmap.getHeight(), CvType.CV_8UC1);
        // Convert the color
        Imgproc.cvtColor(mat, gray, Imgproc.COLOR_RGB2GRAY);
        // Convert back to bitmap
        Mat purpose = new Mat(gray.rows(), gray.cols(), gray.type());

        Imgproc.adaptiveThreshold(gray, purpose, 255, Imgproc.ADAPTIVE_THRESH_MEAN_C, Imgproc.THRESH_BINARY_INV, 255, 25);

        Utils.matToBitmap(purpose, bitmap);
        iv.setImageBitmap(bitmap);
```

```
121
122             Mat m = new Mat();
123             Core.extractChannel(purpose, m, 0);
124             int n = Core.countNonZero(m);
125             int integer = Integer.parseInt(String.valueOf(n));
126             // A4 Area in cm2
127             float a = 623.7f;
128             //Pixel Value A4 Area in 226x320 resolution
129             int r = 72320;
130             double result = (a / r) * integer;
131             Result.setText(String.valueOf(result));
132         }
133     }
134
135
```

Application.java

```
1   package com.example.leafapp;
2
3   import java.io.File;
4   import java.util.List;
5
6   import android.net.Uri;
7   import android.os.Build;
8   import android.os.Bundle;
9   import android.os.Environment;
10  import android.provider.MediaStore;
11  import android.provider.MediaStore.Images;
12  import android.annotation.SuppressLint;
13  import android.content.ContentValues;
14  import android.content.Intent;
15  import android.hardware.Camera;
16  import android.hardware.Camera.CameraInfo;
17  import android.hardware.Camera.Parameters;
18  import android.hardware.Camera.Size;
19  import android.support.v7.app.ActionBarActivity;
20  import android.util.Log;
21  import android.view.Menu;
22  import android.view.MenuItem;
23  import android.view.SubMenu;
24  import android.view.Window;
25  import android.view.WindowManager;
26  import android.widget.Toast;
27
28  import org.opencv.android.BaseLoaderCallback;
29  import org.opencv.android.CameraBridgeViewBase;
30  import org.opencv.android.CameraBridgeViewBase.CvCameraViewFrame;
31  import org.opencv.android.CameraBridgeViewBase.CvCameraViewListener2;
32  import org.opencv.android.LoaderCallbackInterface;
33  import org.opencv.android.JavaCameraView;
34  import org.opencv.android.OpenCVLoader;
35  import org.opencv.core.Core;
36  import org.opencv.core.Mat;
37  import org.opencv.highgui.Highgui;
38  import org.opencv.imgproc.Imgproc;
39
40  // Use the deprecated Camera class.
```

```
 @SuppressWarnings("deprecation")
public final class Application extends ActionBarActivity implements CvCameraViewListener2 {

    // A tag for log output.
    private static final String TAG =
            MainActivity.class.getSimpleName();

    // A key for storing the index of the active camera.
    private static final String STATE_CAMERA_INDEX = "cameraIndex";

    // A key for storing the index of the active image size.
    private static final String STATE_IMAGE_SIZE_INDEX =
            "imageSizeIndex";

    // An ID for items in the image size submenu.
    private static final int MENU_GROUP_ID_SIZE = 2;

    // The index of the active camera.
    private int mCameraIndex;

    // The index of the active image size.
    private int mImageSizeIndex;

    // Whether the active camera is front-facing.
    // If so, the camera view should be mirrored.
    private boolean mIsCameraFrontFacing;

    // The number of cameras on the device.
    private int mNumCameras;

    // The image sizes supported by the active camera.
    private List<Size> mSupportedImageSizes;

    // The camera view.
    private CameraBridgeViewBase mCameraView;

    // Whether the next camera frame should be saved as a photo.
    private boolean mIsPhotoPending;

    // A matrix that is used when saving photos.
```

```
81      private Mat mBgr;
82
83      // Whether an asynchronous menu action is in progress.
84      // If so, menu interaction should be disabled.
85      private boolean mIsMenuLocked;
86
87      // The OpenCV loader callback.
88      private BaseLoaderCallback mLoaderCallback =
89              new BaseLoaderCallback(this) {
90                  @Override
91                  public void onManagerConnected(final int status) {
92                      switch (status) {
93                          case LoaderCallbackInterface.SUCCESS:
94                              Log.d(TAG, "OpenCV loaded successfully");
95                              mCameraView.enableView();
96                              //mCameraView.enableFpsMeter();
97                              mBgr = new Mat();
98                              break;
99                          default:
100                             super.onManagerConnected(status);
101                             break;
102                     }
103                 }
104             };
105
106     // Suppress backward incompatibility errors because we provide
107     // backward-compatible fallbacks.
108     @SuppressLint("NewApi")
109     @Override
110     protected void onCreate(final Bundle savedInstanceState) {
111         super.onCreate(savedInstanceState);
112
113         final Window window = getWindow();
114         window.addFlags(
115                 WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON);
116
117         if (savedInstanceState != null) {
118             mCameraIndex = savedInstanceState.getInt(
119                     STATE_CAMERA_INDEX, 0);
120             mImageSizeIndex = savedInstanceState.getInt(
```

```
121                     STATE_IMAGE_SIZE_INDEX, 0);
122             } else {
123                 mCameraIndex = 0;
124                 mImageSizeIndex = 0;
125             }
126 
127             final Camera camera;
128             if (Build.VERSION.SDK_INT >=
129                     Build.VERSION_CODES.GINGERBREAD) {
130                 CameraInfo cameraInfo = new CameraInfo();
131                 Camera.getCameraInfo(mCameraIndex, cameraInfo);
132                 mIsCameraFrontFacing =
133                         (cameraInfo.facing ==
134                                 CameraInfo.CAMERA_FACING_FRONT);
135                 mNumCameras = Camera.getNumberOfCameras();
136                 camera = Camera.open(mCameraIndex);
137             } else { // pre-Gingerbread
138                 // Assume there is only 1 camera and it is rear-facing.
139                 mIsCameraFrontFacing = false;
140                 mNumCameras = 1;
141                 camera = Camera.open();
142             }
143             final Parameters parameters = camera.getParameters();
144             camera.release();
145             mSupportedImageSizes =
146                     parameters.getSupportedPreviewSizes();
147             final Size size = mSupportedImageSizes.get(mImageSizeIndex);
148 
149             mCameraView = new JavaCameraView(this, mCameraIndex);
150             mCameraView.setMaxFrameSize(size.width, size.height);
151             mCameraView.setCvCameraViewListener(this);
152             setContentView(mCameraView);
153         }
154 
155         public void onSaveInstanceState(Bundle savedInstanceState) {
156             // Save the current camera index.
157             savedInstanceState.putInt(STATE_CAMERA_INDEX, mCameraIndex);
158 
159             // Save the current image size index.
160             savedInstanceState.putInt(STATE_IMAGE_SIZE_INDEX,
```

```
161                 mImageSizeIndex);
162
163         super.onSaveInstanceState(savedInstanceState);
164     }
165
166     // Suppress backward incompatibility errors because we provide
167     // backward-compatible fallbacks.
168     @SuppressLint("NewApi")
169     @Override
170     public void recreate() {
171         if (Build.VERSION.SDK_INT >=
172                 Build.VERSION_CODES.HONEYCOMB) {
173             super.recreate();
174         } else {
175             finish();
176             startActivity(getIntent());
177         }
178     }
179
180     @Override
181     public void onPause() {
182         if (mCameraView != null) {
183             mCameraView.disableView();
184         }
185         super.onPause();
186     }
187
188     @Override
189     public void onResume() {
190         super.onResume();
191         OpenCVLoader.initAsync(OpenCVLoader.OPENCV_VERSION_2_4_9,
192                 this, mLoaderCallback);
193         mIsMenuLocked = false;
194     }
195
196     @Override
197     public void onDestroy() {
198         if (mCameraView != null) {
199             mCameraView.disableView();
200         }
```

```
201         super.onDestroy();
202     }
203
204     @Override
205     public boolean onCreateOptionsMenu(final Menu menu) {
206         getMenuInflater().inflate(R.menu.activity_camera, menu);
207         int numSupportedImageSizes = mSupportedImageSizes.size();
208         if (numSupportedImageSizes > 1) {
209             final SubMenu sizeSubMenu = menu.addSubMenu(R.string.menu_image_size);
210             for (int i = 0; i < numSupportedImageSizes; i++) {
211                 final Size size = mSupportedImageSizes.get(i);
212                 sizeSubMenu.add(MENU_GROUP_ID_SIZE, i, Menu.NONE,
213                         String.format("%dx%d", size.width,
214                                 size.height));
215             }
216         }
217         return true;
218     }
219
220     // Suppress backward incompatibility errors because we provide
221     // backward-compatible fallbacks (for recreate).
222     @SuppressLint("NewApi")
223     @Override
224     public boolean onOptionsItemSelected(final MenuItem item) {
225         if (mIsMenuLocked) {
226             return true;
227         }
228         if (item.getGroupId() == MENU_GROUP_ID_SIZE) {
229             mImageSizeIndex = item.getItemId();
230             recreate();
231
232             return true;
233         }
234         switch (item.getItemId()) {
235             case R.id.menu_take_photo:
236                 mIsMenuLocked = true;
237
238                 // Next frame, take the photo.
239                 mIsPhotoPending = true;
240
```

```
241                 return true;
242             default:
243                 return super.onOptionsItemSelected(item);
244         }
245     }
246
247     @Override
248     public void onCameraViewStarted(final int width,
249                                     final int height) {
250     }
251
252     @Override
253     public void onCameraViewStopped() {
254     }
255
256     @Override
257     public Mat onCameraFrame(final CvCameraViewFrame inputFrame) {
258         final Mat rgba = inputFrame.rgba();
259
260         if (mIsPhotoPending) {
261             mIsPhotoPending = false;
262             takePhoto(rgba);
263         }
264
265         if (mIsCameraFrontFacing) {
266             // Mirror (horizontally flip) the preview.
267             Core.flip(rgba, rgba, 1);
268         }
269
270         return rgba;
271     }
272
273     private void takePhoto(final Mat rgba) {
274
275         // Determine the path and metadata for the photo.
276         final long currentTimeMillis = System.currentTimeMillis();
277         final String appName = getString(R.string.app_name);
278         final String galleryPath =
279                 Environment.getExternalStoragePublicDirectory(
280                         Environment.DIRECTORY_PICTURES).toString();
```

```
281         final String albumPath = galleryPath + File.separator +
282                 appName;
283         final String photoPath = albumPath + File.separator +
284                 currentTimeMillis + LabActivity.PHOTO_FILE_EXTENSION;
285         final ContentValues values = new ContentValues();
286         values.put(MediaStore.MediaColumns.DATA, photoPath);
287         values.put(Images.Media.MIME_TYPE,
288                 LabActivity.PHOTO_MIME_TYPE);
289         values.put(Images.Media.TITLE, appName);
290         values.put(Images.Media.DESCRIPTION, appName);
291         values.put(Images.Media.DATE_TAKEN, currentTimeMillis);
292
293         // Ensure that the album directory exists.
294         File album = new File(albumPath);
295         if (!album.isDirectory() && !album.mkdirs()) {
296             Log.e(TAG, "Failed to create album directory at " +
297                     albumPath);
298             onTakePhotoFailed();
299             return;
300         }
301
302         // Try to create the photo.
303         Imgproc.cvtColor(rgba, mBgr, Imgproc.COLOR_RGBA2BGR, 3);
304         if (!Highgui.imwrite(photoPath, mBgr)) {
305             Log.e(TAG, "Failed to save photo to " + photoPath);
306             onTakePhotoFailed();
307         }
308         Log.d(TAG, "Photo saved successfully to " + photoPath);
309
310         // Try to insert the photo into the MediaStore.
311         Uri uri;
312         try {
313             uri = getContentResolver().insert(
314                     Images.Media.EXTERNAL_CONTENT_URI, values);
315         } catch (final Exception e) {
316             Log.e(TAG, "Failed to insert photo into MediaStore");
317             e.printStackTrace();
318
319             // Since the insertion failed, delete the photo.
320             File photo = new File(photoPath);
```

```
322                 Log.e(TAG, "Failed to delete non-inserted photo");
323             }
324
325             onTakePhotoFailed();
326             return;
327         }
328
329         // Open the photo in LabActivity.
330         final Intent intent = new Intent(this, LabActivity.class);
331         intent.putExtra(LabActivity.EXTRA_PHOTO_URI, uri);
332         intent.putExtra(LabActivity.EXTRA_PHOTO_DATA_PATH,
333                 photoPath);
334         runOnUiThread(new Runnable() {
335             @Override
336             public void run() {
337                 startActivity(intent);
338             }
339         });
340     }
341
342     private void onTakePhotoFailed() {
343         mIsMenuLocked = false;
344
345         // Show an error message.
346         final String errorMessage =
347                 getString(R.string.photo_error_message);
348         runOnUiThread(new Runnable() {
349             @Override
350             public void run() {
351                 Toast.makeText(Application.this, errorMessage,
352                         Toast.LENGTH_SHORT).show();
353             }
354         });
355     }
356 }
```

LabActivity.java

```java
package com.example.leafapp;

import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.provider.MediaStore;
import android.provider.MediaStore.Images;
import android.support.v7.app.ActionBarActivity;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.ImageView;

public final class LabActivity extends ActionBarActivity {

    public static final String PHOTO_FILE_EXTENSION = ".png";
    public static final String PHOTO_MIME_TYPE = "image/png";

    public static final String EXTRA_PHOTO_URI =
            "com.nummist.secondsight.LabActivity.extra.PHOTO_URI";
    public static final String EXTRA_PHOTO_DATA_PATH =
            "com.nummist.secondsight.LabActivity.extra.PHOTO_DATA_PATH";

    private Uri mUri;
    private String mDataPath;

    @Override
    protected void onCreate(final Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        final Intent intent = getIntent();
        mUri = intent.getParcelableExtra(EXTRA_PHOTO_URI);
        mDataPath = intent.getStringExtra(EXTRA_PHOTO_DATA_PATH);

        final ImageView imageView = new ImageView(this);
        imageView.setImageURI(mUri);

        setContentView(imageView);
    }
```

```
41
42      @Override
43      public boolean onCreateOptionsMenu(final Menu menu) {
44          getMenuInflater().inflate(R.menu.activity_lab, menu);
45          return true;
46      }
47
48      @Override
49      public boolean onOptionsItemSelected(final MenuItem item) {
50          switch (item.getItemId()) {
51              case R.id.menu_delete:
52                  deletePhoto();
53                  return true;
54              case R.id.menu_edit:
55                  editPhoto();
56                  return true;
57              default:
58                  return super.onOptionsItemSelected(item);
59          }
60      }
61
62      /*
63       * Show a confirmation dialog. On confirmation ("Delete"), the
64       * photo is deleted and the activity finishes.
65       */
66      private void deletePhoto() {
67          final AlertDialog.Builder alert = new AlertDialog.Builder(
68                  LabActivity.this);
69          alert.setTitle(R.string.photo_delete_prompt_title);
70          alert.setMessage(R.string.photo_delete_prompt_message);
71          alert.setCancelable(false);
72          alert.setPositiveButton(R.string.delete,
73                  new DialogInterface.OnClickListener() {
74                      @Override
75                      public void onClick(final DialogInterface dialog,
76                                          final int which) {
77                          getContentResolver().delete(
78                                  Images.Media.EXTERNAL_CONTENT_URI,
79                                  MediaStore.MediaColumns.DATA + "=?",
80                                  new String[]{mDataPath});
```

```
                    finish();
                }
            });
        alert.setNegativeButton(android.R.string.cancel, null);
        alert.show();
    }

    /*
     * Show a chooser so that the user may pick an app for editing
     * the photo.
     */
    private void editPhoto() {
        final Intent intent = new Intent(Intent.ACTION_EDIT);
        intent.setDataAndType(mUri, PHOTO_MIME_TYPE);
        startActivity(Intent.createChooser(intent,
                getString(R.string.photo_edit_chooser_title)));
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        finish();
    }
}
```

Help.java

```
package com.example.leafapp;

import android.app.Activity;
import android.os.Bundle;

public class Help extends Activity{

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.help);
    }
}

```

XML

Layout

activity_main.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.leafapp_50.MainActivity"
    android:background="#ffffff">

    <ImageView
        android:layout_width="400px"
        android:layout_height="350px"
        android:id="@+id/imageView"
        android:layout_alignParentTop="false"
        android:layout_centerHorizontal="true" />


    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="110dp"
        android:layout_height="20dp"
        android:gravity="bottom"
        android:id="@+id/linearLayout1"
        android:layout_below="@+id/imageView"
        android:layout_centerHorizontal="true">

        <TextView
            android:id="@+id/source"
            android:layout_width="50dp"
            android:layout_height="20dp"
            android:textStyle="bold"
            android:textColor="#000000" />

        <TextView
        android:layout_width="50dp"
        android:layout_height="wrap_content"
        android:text="cm2"
```

```
        android:id="@+id/unit"
        android:layout_centerHorizontal="true"
            android:layout_marginLeft="10dp"
            android:textColor="#000000"
            android:textStyle="bold" />
    </LinearLayout>
    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:gravity="bottom"
        android:id="@+id/linearLayout"
        android:layout_below="@+id/linearLayout1"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_marginTop="2dp">

        <Button
            android:layout_width="fill_parent"
            android:layout_height="match_parent"
            android:text="Capture"
            android:id="@+id/bCapture"
            android:layout_alignParentBottom="true"
            android:layout_alignLeft="@+id/imageView"
            android:layout_alignStart="@+id/imageView"
            android:layout_weight="1" />

        <Button
            android:layout_width="fill_parent"
            android:layout_height="match_parent"
            android:text="Calc"
            android:id="@+id/bPick"
            android:layout_weight="1" />

        <Button
            android:layout_width="fill_parent"
            android:layout_height="match_parent"
            android:text="Help"
            android:id="@+id/bHelp"
            android:layout_weight="1" />
```

```

    </LinearLayout>

</RelativeLayout>
```

leaf.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/cover"
    >


</LinearLayout>
```

help.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:weightSum="1">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:text="Petunjuk Penggunaan Aplikasi"
        android:id="@+id/textView7"
        android:layout_gravity="center_horizontal"
        android:layout_weight="0.03" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="1. Tekan tombol &quot;Capture&quot; untuk mengambil citra                                    "
        android:id="@+id/textView"
        android:layout_weight="0.03"
        android:textColor="#000000" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="2. Setelah muncul fungsi kamera atur kamera sedekat mungkin dengan A4 sebagai alas citra
        android:id="@+id/textView2"
        android:layout_weight="0.03"
        android:layout_gravity="center_horizontal"
        android:textColor="#000000" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="3. Kemudian tekan tombol &quot;Take Photo&quot;                    "
        android:id="@+id/textView3"
        android:layout_weight="0.03"
        android:layout_gravity="center_horizontal"
```

```
        android:textColor="#000000" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="4. Kemudian akan muncul pilihan &quot;Delete&quot; dan &quot; Edit&quot;. Pilih &quot;Delete&quot;
        jika ingin mengambil citra lagi dan Pilih &quot;Edit&quot; untuk memotong citra"
        android:id="@+id/textView4"
        android:layout_weight="0.03"
        android:layout_gravity="center_horizontal"
        android:textColor="#000000" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="5. Pilih &quot;Save&quot; untuk menyimpan citra                  "
        android:id="@+id/textView5"
        android:layout_weight="0.03"
        android:textColor="#000000" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="6. Tekan tombol &quot;Calc&quot; untuk memilih citra yang akan diukur                                  "
        android:id="@+id/textView6"
        android:layout_weight="0.03"
        android:layout_gravity="center_horizontal"
        android:textColor="#000000" />

</LinearLayout>
```

Menu

activity_camera.xml

```
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">
    <item
        android:id="@+id/menu_take_photo"
        app:showAsAction="ifRoom|withText"
        android:title="@string/menu_take_photo" />
</menu>
```

activity_lab.xml

```
<menu
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">
    <item
        android:id="@+id/menu_delete"
        app:showAsAction="ifRoom|withText"
        android:title="@string/delete" />
    <item
        android:id="@+id/menu_edit"
        app:showAsAction="ifRoom|withText"
        android:title="@string/edit" />
    </menu>
```

Value

strings.xml

```xml
<resources>
    <string name="app_name">Leaf App</string>
    <string name="delete">Delete</string>
    <string name="edit">Edit</string>
    <string name="menu_next_camera">Next Cam</string>
    <string name="menu_take_photo">Take Photo</string>
    <string name="menu_image_size">Size</string>
    <string name="photo_delete_prompt_message">This photo is saved in your Gallery. Do you want to delete it?</string>
    <string name="photo_delete_prompt_title">Delete photo?</string>
    <string name="photo_error_message">Failed to save photo</string>
    <string name="photo_edit_chooser_title">Edit photo with&#8230;</string>
    <string name="photo_send_chooser_title">Share photo with&#8230;</string>
    <string name="photo_send_extra_subject">My photo from Second Sight</string>
    <string name="photo_send_extra_text">Check out my photo from the Second Sight app! http://nummist.com/opencv/</string>
    <string name="share">Share</string>
</resources>

```

AndroidManifest.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.leafapp">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/icon_2"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".Cover">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="com.example.leafapp.MAINACTIVITY" />
                <category android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>
        <activity
            android:name=".Application"
            android:screenOrientation="landscape">
        </activity>
        <activity
            android:name="com.example.leafapp.LabActivity"
            android:label="@string/app_name"
            android:screenOrientation="landscape">
        </activity>
        <activity
            android:name=".Help">
        </activity>
    </application>

    <uses-permission android:name="android.permission.CAMERA"/>
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
    <uses-feature android:name="android.hardware.camera"/>
    <uses-feature android:name="android.hardware.camera.autofocus"
```

```
41          android:required="false"/>
42      <uses-feature android:name="android.hardware.camera.flash"
43          android:required="false"/>
44
45  </manifest>
46
```