Information of the InsDet Dataset

1. Data Description

This section introduces images of objects and scenes taken by a cellphone that is equipped with a Leica camera.

Our dataset is available on the github repository: https://github.com/insdet/instance-detection

InsDet structure:

- Objects
- Scenes
- Background

1.1 Raw data

The camera settings are illustrated in the table below.

object_id: 000~040, 052~092		**object_id: 041~051, 093~099**	
3072×3072 pixels	ISO: 50 shutter speed: 1/160 optical zoom: 0.7x simulated aperture: f/1.9 focus length: 2mm	**3456×3456** pixels	ISO: 125 shutter speed: 1/160 optical zoom: 1x simulated aperture: f/2.2 focus length: 2mm

The “**Objects**” folder contains:

- For each of (currently) **100 objects** (object_name), we provide 1 sampling view (45° to object center) and 24 rotation position (24 total)
    - The first three digits specify the id of each object. (e.g. 000_aveda_shampoo)
    - raw RGB image (e.g. “images/001.jpg”)
    - segmentation masks for the RGB images (e.g. “masks/001.png”) generated by GrabCut Annotation toolbox

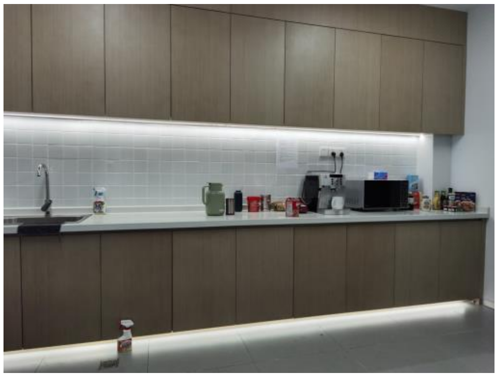
The “**Scenes**” folder contains:

- For each of (currently) **14 test scenes** with **160** testing images in total, we provide images with different angles of sampling views and different lighting environments.
    - raw RGB images with 6144×8192 pixels (e.g. “office001/rgb_000.jpg”)

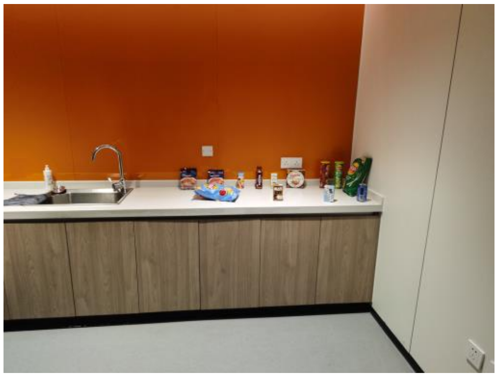
- Bounding box annotation for objects in test scenes (e.g. “office001/rgb_000.xml”) generated by labelImg toolbox and using PascalVOC format.

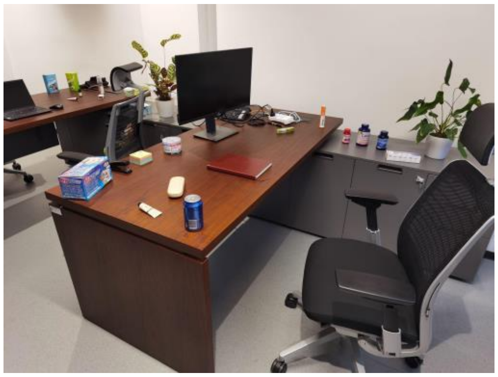
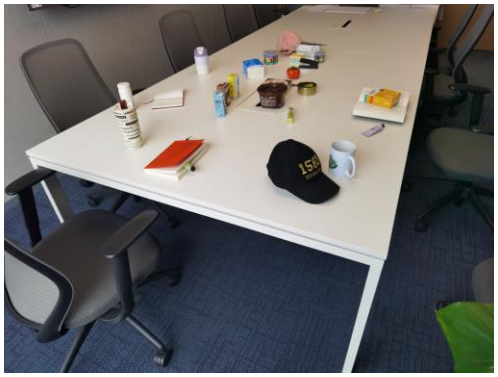
- Each bounding box is specified by [xmin, ymin, xmax, ymax].

- **Hard** scenes: office_001, pantry_room_001

- **Easy** scenes: leisure_zone, meeting_room, office_002, pantry_room_002, sink

Object categories

- “Objects” appear in “Scenes”
  - **3072×3072 (000~040, 052~092)**

000_aveda_shampoo
001_binder_clips_median
002_binder_clips_small
003_bombik_bucket
004_bonne_maman_blueberry
005_bonne_maman_raspberry
006_bonne_maman_strawberry

052_acnes_cream
053_aveda_conditioner
054_banana_milk_drink
055_candle_beast
056_china_persimmon
057_danisa_butter_cookies
058_effaclar_duo

```
007_costa_caramel
008_essential_oil_bergamot
009_garlic_toast_spread
010_handcream_avocado
011_hb_calcium
012_hb_grapeseed
013_hb_marine_collagen
014_hellmanns_mayonnaise
015_illy_blend
016_japanese_finger_cookies
017_john_west_canned_tuna
018_kerastase_shampoo
019_kiehls_facial_cream
020_kiihne_balsamic
021_kiihne_honey_mustard
022_lindor_matcha
023_lindor_salted_caramel
024_lush_mask
025_pasta_sauce_black_pepper
026_pasta_sauce_tomato
027_pepsi
028_portable_yogurt_machine
029_selfile_stick
030_sour_lemon_drops
031_sticky_notes
032_stridex_green
033_thermos_flask_cream
034_thermos_flask_muji
035_thermos_flask_sliver
036_tragata_olive_oil
037_tulip_luncheon_meat
038_unicharm_cotton_pad
039_vinda_tissue
040_wrigley_doublemint_gum
059_evelom_cleanser
060_glasses_box_blone
061_handcream_iris
062_handcream_lavender
063_handcream_rosewater
064_handcream_summer_hill
065_hr_serum
066_japanese_chocolate
067_kerastase_hair_treatment
068_kiehls_serum
069_korean_beef_marinade
070_korean_doenjang
071_korean_gochujang
072_korean_ssamjang
073_loccitane_soap
074_marvis_toothpaste_purple
075_mouse_thinkpad
076_oatly_chocolate
077_oatly_original
078_ousa_grated_cheese
079_polaroid_film
080_skinceuticals_be
081_skinceuticals_cf
082_skinceuticals_phyto
083_stapler_black
084_stapler_blue
085_sunscreen_blue
086_tempo_pocket_tissue
087_thermos_flask_purple
088_uha_matcha
089_urban_decay_spray
090_vitaboost_multivitamin
091_watercolor_penbox
092_youthlt_bilberry_complex
```

○ **3456×3456 (041~051, 093~099)**

```
041_baseball_cap_black
042_baseball_cap_pink
043_bfe_facial_mask
044_corgi_doll
045_dinosaur_doll
046_geo_mocha
047_geo_roast_charcoal
048_instant_noodle_black
049_instant_noodle_red
050_nabati_cheese_wafer
051_truffettes
093_daiso_mod_remover
094_kaneyo_kitchen_bleach
095_lays_chip_bag_blue
096_lays_chip_bag_green
097_lays_chip_tube_auburn
098_lays_chip_tube_green
099_mug_blue
```

2. Foreground Segmentation -- *GrabCut*

This section introduces the details of how we generate the segmentation masks of objects by using GrabCut algorithm on raw RGB images. The gendata codes are modified from [this publicly available Github repository]().

1. Resize raw images to smaller size (better fit the GrabCut Annotation Graphic Interface)

   e.g. down-sample <lush_mask> from 3072×3072 pixels to 384×384 pixels (8×)

   put lush_mask folder under <base_dir>

   ```
   <base_dir> = ../gendata/data/3072x3072

   python resizeImg.py --datadir <base_dir>/lush_mask/images \

                       --destdir <base_dir>/lush_mask/input \

                       --resolutions 384 384
   ```

2. Segment object manually by using GrabCut interactive algorithm

   ```
   python grabcut.py --input <base_dir>/lush_mask/input \

                     --output <base_dir>/lush_mask/output \

                     --config <base_dir>/config.json
   ```

   Note: The output folder contains an `annotation` folder for segmentation masks and an `image` folder for segmented objects.

3. Resize segmentation masks to original image size

   ```
   python resizeMask.py --datadir <base_dir>/lush_mask/output/annotation \

                        --destdir <base_dir>/lush_mask/masks \

                        --resolutions 3072 3072
   ```

3. Label Annotation -- *labelImg*

For test scenes, we adopt a publicly available annotation toolbox [*labelImg*]() to label the bounding boxes of objects in test scenes. The *predefine_classes.txt* has been rewrite and uploaded to the aforementioned Github link. The bounding box of an object is a rectangle that includes the maximum area of the object appearing in the test scene.

4. Data Processing for Cut-Paste-Learn

This section introduces the details of how we process “Objects” data and “Background” data for Cut-Paste-Learn. Two Python files `dataset_generator.py` and `defaults.py` in the [cut-paste-learn official github repository]() should be modified for our requirements.

```
<base_dir> = ../InsDet

<dest_dir> = ../syndata-generation/data_dir
```

1. Downsize background images to 4× smaller size

   e.g. original size of 6144×4096 pixels → 768×1024 pixels

   ```
   python resizeImg.py --datadir <base_dir>/Background \
                       --destdir <dest_dir>/background \
                       --resolutions 768 1024
   ```

2. Get bounding box of objects

   ```
   python getBbox.py
   ```

3. Center crop `images` and `masks` to an appropriate size

   ```
   python centerCrop.py
   ```

4. Downsize `images` and `masks` to a smaller size (e.g. 256×256)

   ```
   python resizeImg.py
   ```

   ```
   python resizeMask.py
   ```

5. Invert `masks`（Make background as TRUE and foreground as FALSE), and generate foreground with white background

   ```
   python invertMask.py
   ```

   Note: Related functions of `minify, resizemask, getbbox, centercrop, invertmask` are packed in `datautils.py.` The `loop_resizeImg.py`, `loop_resizeMask.py,` `loop_getBbox.py`, `loop_centerCrop.py`, and `loop_invertMask.py` can be used for a set of folders.

6. Set multiple variables in `defaults.py,` such as (1) the numbers of objects inserted in each background image, (2) the scales of inserted object instances, (3) combination of four commonly-used blending methods, (4) the amounts of synthesized images. Then run `dataset_generator.py` to generate synthesized images.