Product Recognition on Store Shelves

Professor:

Luigi Di Stefano

Students:

Federico Fabbri

Jacopo Merlo Pich

Daniele Simonazzi

Academic year: 2022/2023

Abstract

This report presents the development of a computer vision system for product recognition on store shelves. The system identifies, in a scene image, boxes of cereal of different brands based on a reference image for each product. The proposed algorithm at first extracts keypoints from both images and matches them, and then it filters out the spurious ones thanks to the voting mechanism in a 4D Hough space. Other measures have been introduced to further improve performance.

Contents

Introduction

The aim of this project is to develop a computer vision system that, given a reference image for each product, is able to identify boxes of cereal of different brands from an image of a store shelf.

The project is structured in tasks of increasing difficulty.

In the first part we will work with images of shelves in which there are a few boxes. In particular, each individual model appears at most once.

In the second, we add the possibility of observing multiple instances of the same model, albeit always dealing with only a few boxes.

Finally, in the third prompt, the challenge is to find the largest number of boxes in an image containing a typical supermarket shelf full of cereals of all kinds.

The dataset consists of 3 different sets of scenes, each for its own task, and 27 different reference images. It’s worth noting that all of these images have different resolutions and, in many cases, we can observe models that differ only in color.

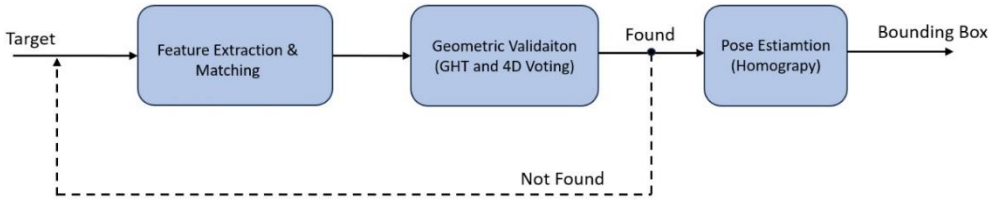
As we aimed to standardize the resolution of the model images across the dataset, we chose to establish a reference area and resize each image while preserving their aspect ratio and, more importantly, their edges. This was accomplished through the use of a median filter.

In order to explain the development of our algorithm, in the next chapters we show the problems that emerged, and the respective solutions adopted.

Task1

Generic approach

The core of the proposed algorithm depends primarily on the use of the object_retrieve function. It can be succinctly encapsulated by the following scheme:

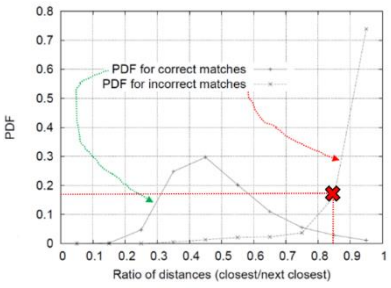


Feature Extraction & Matching

The first part deals with the query (model) image taken into consideration. The keypoints are found with the SIFT detector (DoG), which characterizes each keypoint with a pose, an orientation and a scale. The SIFT descriptor proceeds to compute a 128-element vector that encodes information about the local image gradient orientation and magnitude around each keypoint. We can consider this section as the off-line phase. In fact, these computations are only executed one time for each query image.

The same procedure is used to detect keypoints for the train (scene) image, in the so-called on-line phase. At this point we have keypoints for both query and train images, so the matches can be calculated.

To calculate matches, we depend on the kd-tree algorithm provided by FLANN, which is designed for approximate nearest neighbor search. To ensure the accuracy of our matches, we've employed Lowe's distance ratio test, setting the threshold at 0.85.

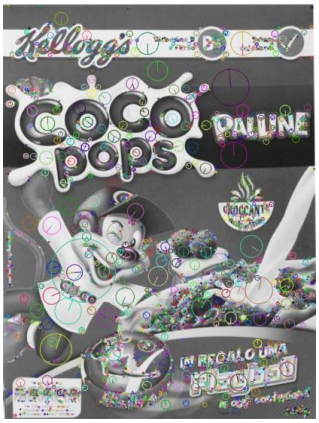


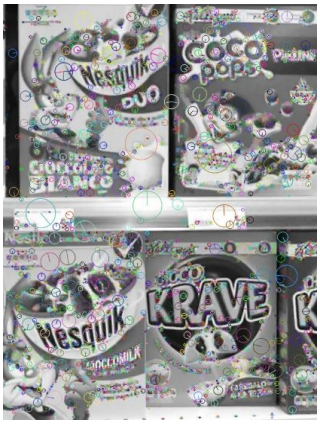
This threshold, slightly higher than the recommended value in existing literature by 0.05. Subsequent filtering further refined the matches, removing incorrect ones.

Here we introduced a threshold **min_match_count** that varies depending on what task we're trying to solve. We selected the threshold values based on some testing and on the behavior of the results that we got, after running the algorithm multiple times. In particular, we found the following values to be the best fit:

- Task A: **min_match_count** = 30
- Task B: **min_match_count** = 10
- Task C: ***min_match_count*** = 4

If the number of "good" matches is not greater than the threshold, the algorithm skips to the next model to be sought, otherwise we proceed with the geometric validation.





Geometric Validation - GHT and 4D Voting

Geometric validation represents a key step for a successful and effective object recognition process. During this phase, the quality of matches is analyzed taking into account geometric considerations and features' geometry to improve the filtering process and validate the obtained results.

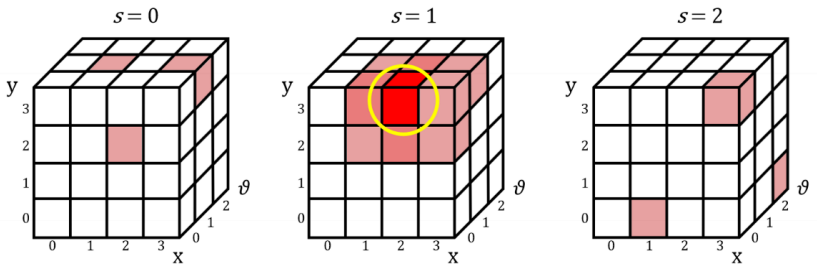
The first step is the off-line phase where the pose of the keypoints is used to compute the barycenter. Then the joining vector between the keypoints and the barycenter are collected into an array.

The second step is the on-line phase where, this time, our goal is to find the barycenter by adding the freshly found joining vectors to the respective keypoints pose. In order to achieve similarity invariance, we compute rotation and scale change:

$$\Delta\varphi_i = \tilde{\varphi}_i - \varphi_i \qquad s_i = \frac{\tilde{S}_i}{S_i}$$

$$\tilde{P}_{C_i} = \tilde{P}_i + s_i \cdot R \cdot (\Delta\varphi_i) \cdot V_i$$

Finally, we can proceed to the voting process. In particular we chose to define a quantized 4D Hough space in which translation, rotation and scale are divided into bins.



Each correspondence provides a triplet. Coherent correspondence will cast votes into the very same bin. Thus, peaks in the 4D space highlight the most likely pose hypotheses concerning the sought object. Peaks are then thresholded to decide whether each such hypothesis should be accepted or rejected.

For sake of completeness, we report below the values that have been empirically selected:

- 10 bins for x and y
- 10 bins for the $\Delta\varphi_i$
- 20 bins for the scaling

In order to identify the position of the barycenter, we implemented a grid over the scene image. The voting part is done by looking at which grid cell is the one where most of the joining vector point, so the one where is most likely for the baricentre position to be. We opted for a consistent number of grid rows and columns across all three tasks. This decision is driven by the nature of the tasks. In the first task, which involves a limited number of boxes, our primary goal is to effectively filter out any erroneous matches. Conversely, in the last task, as we will explain in the upcoming chapter, the training images have lower resolution and consequently exhibit very few

matches. Applying a stringent filtering approach in this case would result in false rejections and a decrease in object detection performance.

Pose Estimation

Given the set of "really good" matches obtained as the output of the voting process, we conclude by estimating the position of the query object in the train image. This step relies on the estimation of an homography based on the correspondences. Ones this transformation matrix is obtained we use it to re-map the corner in the query image into the corresponding point of the train image.

Problems

Color Problem

During the development of this task, two main problems were encountered. The first is related to the fact that SIFT works with black and white images. As can be seen in the image below there are models in the dataset that are very similar in pattern but different in color. Hence in all these cases may not be able to choose correctly.

The implemented solution proposes to work in Hue-Saturation-Value color space. The HSV color space is often used in applications where color manipulation, selection, or detection is important. It offers advantages over the RGB color model because it separates the color information (hue and saturation) from the brightness information (value), making it easier to adjust and work with colors in a more intuitive manner.

To identify the color of the analyzed box, we establish a range of hues and then examine the image to gauge the degree to which that specific color range is represented.

Recognition ambiguity

Another issue we faced was the situation where multiple models were linked to the same box within the train image. Indeed, as can be seen in the following example, model/0 and model/26 were both detected due to their similarity.

model/0

model/26

Because, as expected, the incorrect query image yielded a significantly lower number of matches, we opted to base our final decision on the match count by selecting the higher one.

Results for task 1

RESULT FOR: scenes/e1.png

models/0.jpg - 1 instances found:

Instance 1: - position: [162, 216] - width: 310px - height: 432px

models/11.jpg - 1 instances found:

Instance 1: - position: [442, 182] - width: 301px - height: 364px

RESULT FOR: scenes/e2.png

models/24.jpg - 1 instances found:

Instance 1: - position: [167, 232] - width: 335px - height: 464px

models/26.jpg - 1 instances found:

Instance 1: - position: [538, 231] - width: 334px - height: 462px

models/25.jpg - 1 instances found:

Instance 1: - position: [878, 232] - width: 313px - height: 442px

RESULT FOR: scenes/e3.png

models/0.jpg - 1 instances found:

Instance 1: - position: [170, 236] - width: 325px - height: 439px

models/1.jpg - 1 instances found:

Instance 1: - position: [818, 197] - width: 307px - height: 394px

models/11.jpg - 1 instances found:

Instance 1: - position: [474, 192] - width: 303px - height: 384px

Task2

The solution of this task follows the generic approach we illustrated earlier except for a few necessary adjustments we had to make.

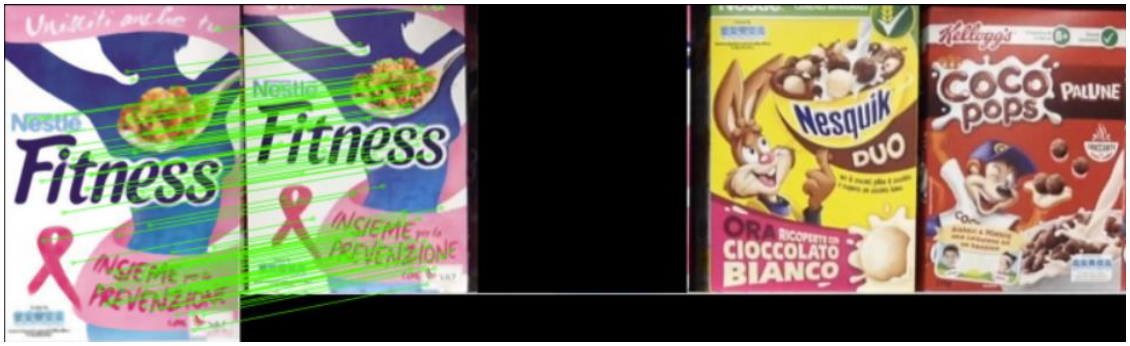
Multi-instances handling

The main problem we encountered was the presence of multi-instances of the same model image. If we were to repeat the previous algorithm as is, we would find ourselves stuck in a continuous loop of the same object detection. This is likely due to the robustness of the FLANN-based matcher, which consistently identifies matches belonging to the same object within the scene. To address this issue, we devised a straightforward yet effective solution. We continued to iterate the algorithm but with a crucial modification: we removed the region of interest (ROI) in which the previous instance of the model had been detected.

It's important to note that this "patch" is exclusively applied to a copy of the scene image, and any alterations made to it will persist until the next model is being sought. This is because, as mentioned earlier, the algorithm may identify different models within the same ROI. By applying the patch, we would effectively restrict our ability to search for other models in that area. Additionally, there is a possibility of incorrect matches and subsequently incorrect homographies, which could significantly alter the train image and ultimately cause the algorithm to fail.

1) Find matches for the first box

2) Apply the patch over the box just found



3) Find matches for the second instance of the box

4) Apply the patch over the second box just found

Results

Once more, the results showcase the effectiveness of the solutions we have proposed.

```
RESULT FOR:  scenes/m1.png


models/24.jpg - 2 instances found:
            Instance 1: - position: [184, 232] - width: 352px - height: 464px
            Instance 2: - position: [550, 232] - width: 336px - height: 464px
models/26.jpg - 1 instances found:
            Instance 1: - position: [921, 231] - width: 334px - height: 462px
models/25.jpg - 1 instances found:
            Instance 1: - position: [1263, 288] - width: 334px - height: 564px
```


```
RESULT FOR:  scenes/m2.png


models/0.jpg - 1 instances found:
            Instance 1: - position: [178, 300] - width: 343px - height: 448px
models/1.jpg - 2 instances found:
            Instance 1: - position: [848, 258] - width: 312px - height: 412px
            Instance 2: - position: [1180, 254] - width: 302px - height: 407px
models/11.jpg - 1 instances found:
            Instance 1: - position: [510, 253] - width: 307px - height: 403px
```

RESULT FOR: scenes/m3.png

models/19.jpg - 1 instances found:

Instance 1: - position: [1233, 190] - width: 296px - height: 381px

models/26.jpg - 1 instances found:

Instance 1: - position: [195, 230] - width: 355px - height: 460px

models/25.jpg - 2 instances found:

Instance 1: - position: [558, 225] - width: 336px - height: 451px

Instance 2: - position: [897, 292] - width: 350px - height: 571px

RESULT FOR: scenes/m4.png

models/24.jpg - 2 instances found:

Instance 1: - position: [539, 223] - width: 349px - height: 446px

Instance 2: - position: [171, 223] - width: 343px - height: 446px

models/26.jpg - 1 instances found:

Instance 1: - position: [921, 221] - width: 342px - height: 443px

models/25.jpg - 2 instances found:

Instance 1: - position: [1271, 216] - width: 324px - height: 433px

Instance 2: - position: [1588, 214] - width: 308px - height: 428px

RESULT FOR: scenes/m5.png

models/1.jpg - 2 instances found:

Instance 1: - position: [500, 720] - width: 312px - height: 410px

Instance 2: - position: [833, 716] - width: 304px - height: 407px

models/11.jpg - 1 instances found:

Instance 1: - position: [161, 723] - width: 315px - height: 347px

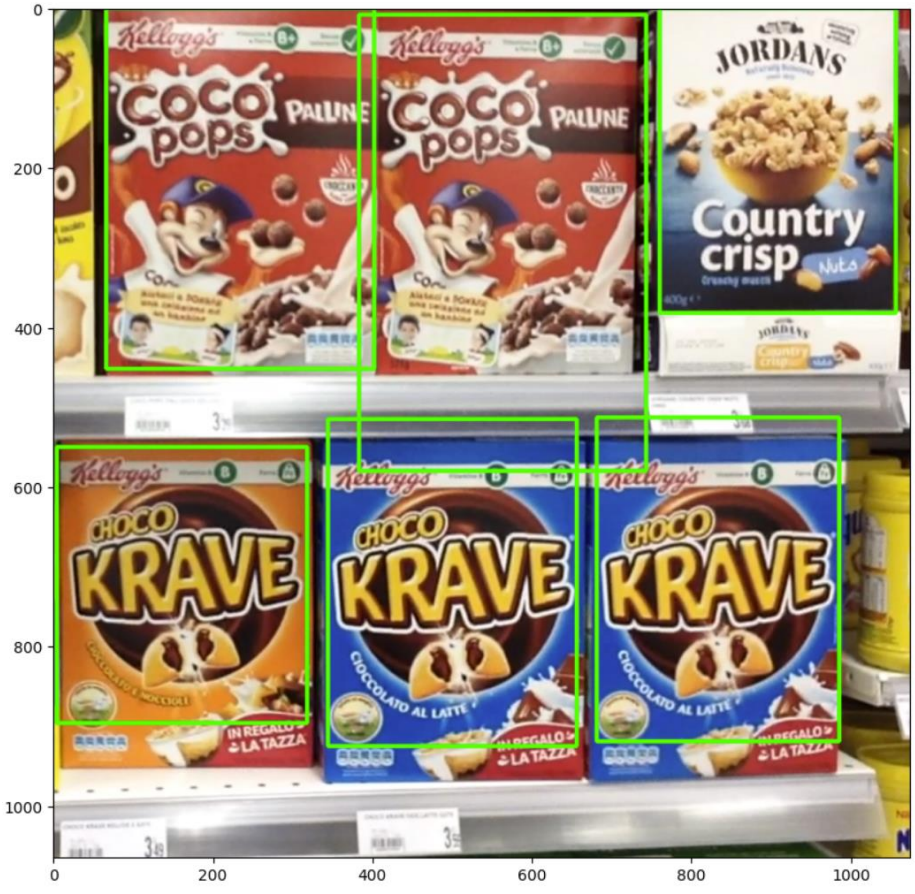
models/19.jpg - 1 instances found:

Instance 1: - position: [908, 191] - width: 297px - height: 382px

models/25.jpg - 2 instances found:

Instance 1: - position: [563, 294] - width: 360px - height: 572px

Instance 2: - position: [234, 226] - width: 336px - height: 453px



Task3 (Optional)

This final task certainly takes on the aspect of a challenge. Indeed, these images contain more than 40 different product instances in each picture, in addition to being of low resolution and featuring numerous distracting elements.

The real challenge lies in finding a balance between identifying as many matches as possible while ensuring their quality through effective filtering.

This is why, in this particular case, we opted for a less strict grid and set the minimum number of final matches at 4, which also happens to be the minimum required to obtain the homography matrix.

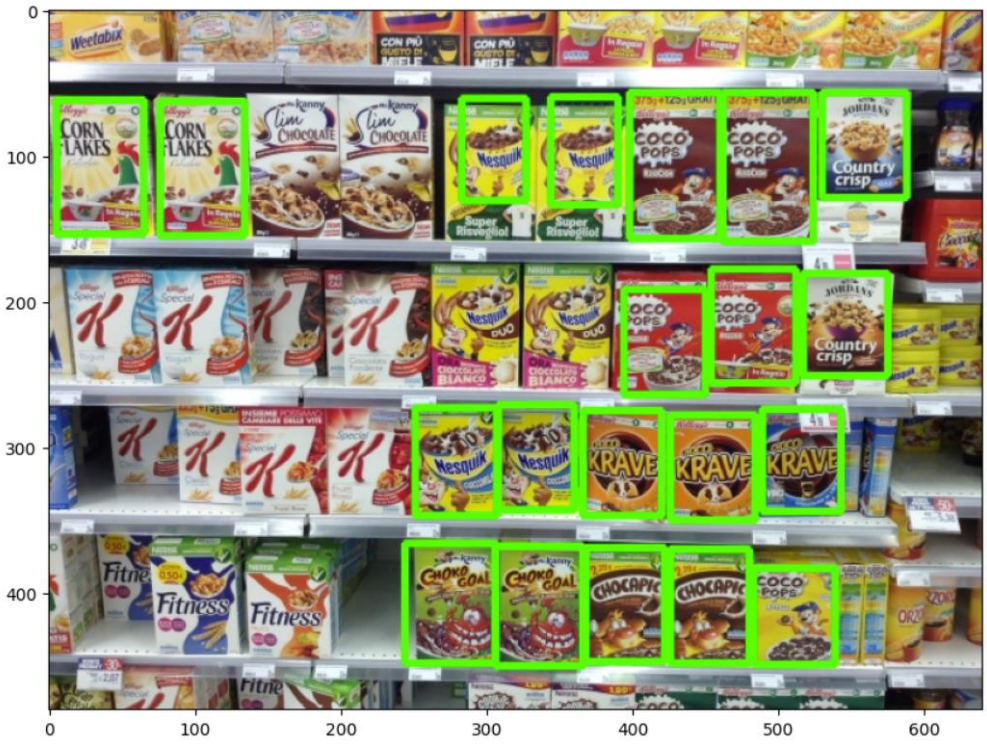
Results

As a consequence of having these multiple filters in place, the number of correctly detected objects stays steady throughout all the train images, typically yielding approximately 20 objects detected per image. It's important to reiterate that the less stringent we are during the filtering phase, the greater the likelihood of incorrect detections. Therefore, we are content with the current setup as it strikes a balance between accuracy and comprehensiveness.

```
RESULT FOR:  scenes/h1.jpg

models/0.jpg - 2 instances found:
           Instance 1: - position: [279, 310] - width: 57px - height: 74px
           Instance 2: - position: [334, 307] - width: 53px - height: 73px
models/1.jpg - 1 instances found:
           Instance 1: - position: [516, 309] - width: 55px - height: 71px
models/2.jpg - 2 instances found:
           Instance 1: - position: [35, 108] - width: 61px - height: 92px
           Instance 2: - position: [105, 109] - width: 59px - height: 92px
models/3.jpg - 2 instances found:
           Instance 1: - position: [275, 407] - width: 63px - height: 83px
           Instance 2: - position: [336, 409] - width: 60px - height: 83px
```

```
models/5.jpg - 2 instances found:
            Instance 1: - position: [367, 97] - width: 46px - height: 75px
            Instance 2: - position: [305, 96] - width: 45px - height: 71px
models/7.jpg - 2 instances found:
            Instance 1: - position: [429, 107] - width: 63px - height: 101px
            Instance 2: - position: [493, 108] - width: 63px - height: 102px
models/8.jpg - 2 instances found:
            Instance 1: - position: [422, 228] - width: 57px - height: 73px
            Instance 2: - position: [483, 217] - width: 59px - height: 79px
models/11.jpg - 2 instances found:
            Instance 1: - position: [455, 314] - width: 59px - height: 73px
            Instance 2: - position: [393, 311] - width: 54px - height: 73px
models/14.jpg - 2 instances found:
            Instance 1: - position: [453, 409] - width: 54px - height: 79px
            Instance 2: - position: [395, 408] - width: 55px - height: 79px
models/15.jpg - 1 instances found:
            Instance 1: - position: [511, 416] - width: 57px - height: 66px
models/18.jpg - 1 instances found:
            Instance 1: - position: [546, 216] - width: 59px - height: 70px
models/19.jpg - 1 instances found:
            Instance 1: - position: [559, 93] - width: 58px - height: 73px
```



Result for scenes/h1.jpg

RESULT FOR: scenes/h2.jpg

models/0.jpg - 2 instances found:

Instance 1: - position: [310, 325] - width: 56px - height: 68px

Instance 2: - position: [252, 324] - width: 56px - height: 71px

models/2.jpg - 1 instances found:

Instance 1: - position: [85, 113] - width: 61px - height: 118px

models/3.jpg - 3 instances found:

Instance 1: - position: [363, 421] - width: 62px - height: 79px

Instance 2: - position: [305, 424] - width: 56px - height: 77px

Instance 3: - position: [248, 422] - width: 58px - height: 79px

models/7.jpg - 2 instances found:

Instance 1: - position: [393, 129] - width: 58px - height: 91px

Instance 2: - position: [466, 134] - width: 60px - height: 105px

models/11.jpg - 2 instances found:

Instance 1: - position: [425, 329] - width: 57px - height: 71px

Instance 2: - position: [367, 330] - width: 54px - height: 74px

models/14.jpg - 1 instances found:

Instance 1: - position: [456, 421] - width: 54px - height: 73px

models/15.jpg - 1 instances found:

Instance 1: - position: [514, 426] - width: 54px - height: 62px

models/16.jpg - 2 instances found:

Instance 1: - position: [481, 32] - width: 62px - height: 65px

Instance 2: - position: [542, 34] - width: 61px - height: 68px

models/18.jpg - 1 instances found:

Instance 1: - position: [507, 235] - width: 55px - height: 67px

models/19.jpg - 1 instances found:

Instance 1: - position: [533, 133] - width: 57px - height: 70px

models/22.jpg - 2 instances found:

Instance 1: - position: [223, 32] - width: 63px - height: 64px

Instance 2: - position: [289, 32] - width: 63px - height: 65px

Result for scenes/h2.jpg

RESULT FOR: scenes/h3.jpg

models/0.jpg - 2 instances found:

Instance 1: - position: [404, 311] - width: 55px - height: 72px

Instance 2: - position: [345, 309] - width: 55px - height: 71px

models/1.jpg - 1 instances found:

Instance 1: - position: [586, 318] - width: 59px - height: 76px

models/2.jpg - 1 instances found:

Instance 1: - position: [84, 378] - width: 49px - height: 55px

models/3.jpg - 3 instances found:

Instance 1: - position: [456, 410] - width: 59px - height: 79px

Instance 2: - position: [399, 408] - width: 59px - height: 77px

Instance 3: - position: [342, 404] - width: 55px - height: 77px

models/4.jpg - 2 instances found:

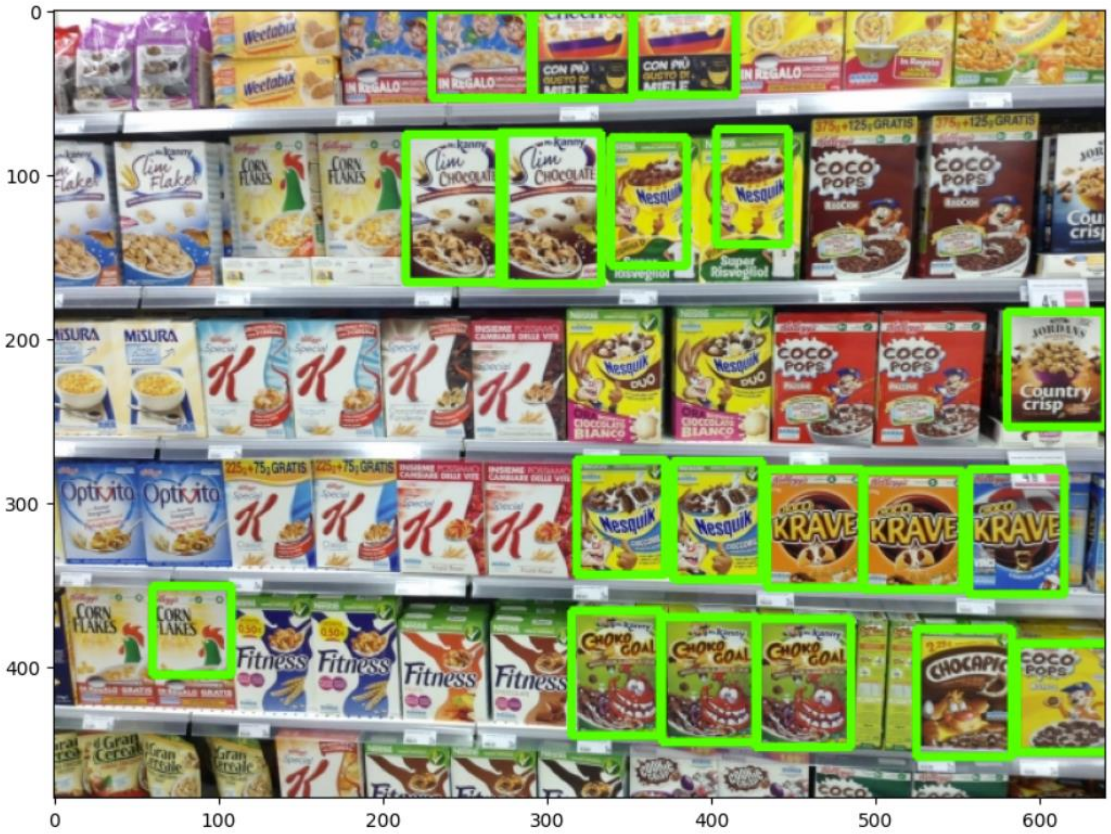
Instance 1: - position: [302, 121] - width: 62px - height: 93px

Instance 2: - position: [244, 121] - width: 61px - height: 91px

models/5.jpg - 2 instances found:

Instance 1: - position: [362, 117] - width: 46px - height: 79px

```
            Instance 2: - position: [425, 108] - width: 45px - height: 70px
models/11.jpg - 2 instances found:
            Instance 1: - position: [523, 317] - width: 60px - height: 72px
            Instance 2: - position: [463, 316] - width: 58px - height: 72px
models/14.jpg - 1 instances found:
            Instance 1: - position: [554, 416] - width: 59px - height: 78px
models/15.jpg - 1 instances found:
            Instance 1: - position: [615, 419] - width: 59px - height: 66px
models/18.jpg - 1 instances found:
            Instance 1: - position: [610, 219] - width: 60px - height: 70px
models/21.jpg - 1 instances found:
            Instance 1: - position: [260, 27] - width: 60px - height: 55px
models/22.jpg - 2 instances found:
            Instance 1: - position: [322, 27] - width: 61px - height: 55px
            Instance 2: - position: [384, 26] - width: 62px - height: 53px
```



Result for scenes/h3.jpg

RESULT FOR: scenes/h4.jpg

models/0.jpg - 1 instances found:

Instance 1: - position: [327, 337] - width: 59px - height: 78px

models/1.jpg - 1 instances found:

Instance 1: - position: [507, 334] - width: 58px - height: 72px

models/2.jpg - 2 instances found:

Instance 1: - position: [85, 130] - width: 64px - height: 96px

Instance 2: - position: [23, 131] - width: 46px - height: 95px

models/3.jpg - 2 instances found:

Instance 1: - position: [266, 441] - width: 63px - height: 89px

Instance 2: - position: [328, 441] - width: 63px - height: 87px

models/5.jpg - 2 instances found:

Instance 1: - position: [353, 118] - width: 47px - height: 72px

Instance 2: - position: [288, 120] - width: 47px - height: 72px

models/7.jpg - 1 instances found:

Instance 1: - position: [422, 131] - width: 63px - height: 110px

models/8.jpg - 1 instances found:

Instance 1: - position: [457, 246] - width: 53px - height: 70px

models/9.jpg - 1 instances found:

Instance 1: - position: [76, 341] - width: 61px - height: 78px

models/11.jpg - 2 instances found:

Instance 1: - position: [447, 337] - width: 59px - height: 75px

Instance 2: - position: [386, 339] - width: 59px - height: 76px

models/12.jpg - 1 instances found:

Instance 1: - position: [136, 452] - width: 60px - height: 72px

models/14.jpg - 2 instances found:

Instance 1: - position: [388, 436] - width: 58px - height: 83px

Instance 2: - position: [447, 434] - width: 54px - height: 81px

models/15.jpg - 1 instances found:

Instance 1: - position: [504, 438] - width: 55px - height: 67px

models/18.jpg - 1 instances found:

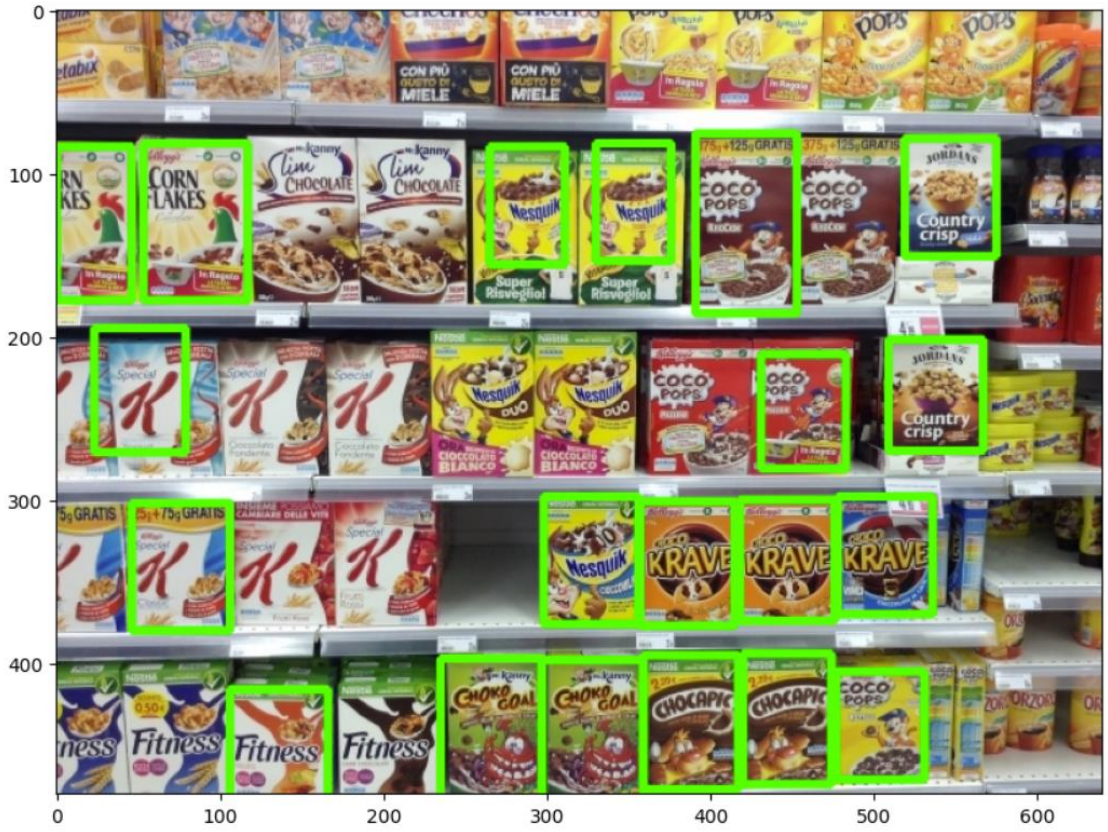
Instance 1: - position: [538, 236] - width: 58px - height: 69px

models/19.jpg - 1 instances found:

Instance 1: - position: [547, 115] - width: 57px - height: 72px

models/20.jpg - 1 instances found:

Instance 1: - position: [51, 233] - width: 57px - height: 77px



Result for scenes/h4.jpg

RESULT FOR: scenes/h5.jpg

models/0.jpg - 2 instances found:

Instance 1: - position: [223, 339] - width: 58px - height: 75px

Instance 2: - position: [284, 339] - width: 58px - height: 75px

models/1.jpg - 1 instances found:

Instance 1: - position: [464, 340] - width: 59px - height: 81px

models/2.jpg - 1 instances found:

Instance 1: - position: [31, 125] - width: 50px - height: 77px

models/3.jpg - 3 instances found:

Instance 1: - position: [220, 440] - width: 61px - height: 81px

Instance 2: - position: [276, 447] - width: 59px - height: 88px

Instance 3: - position: [338, 443] - width: 60px - height: 82px

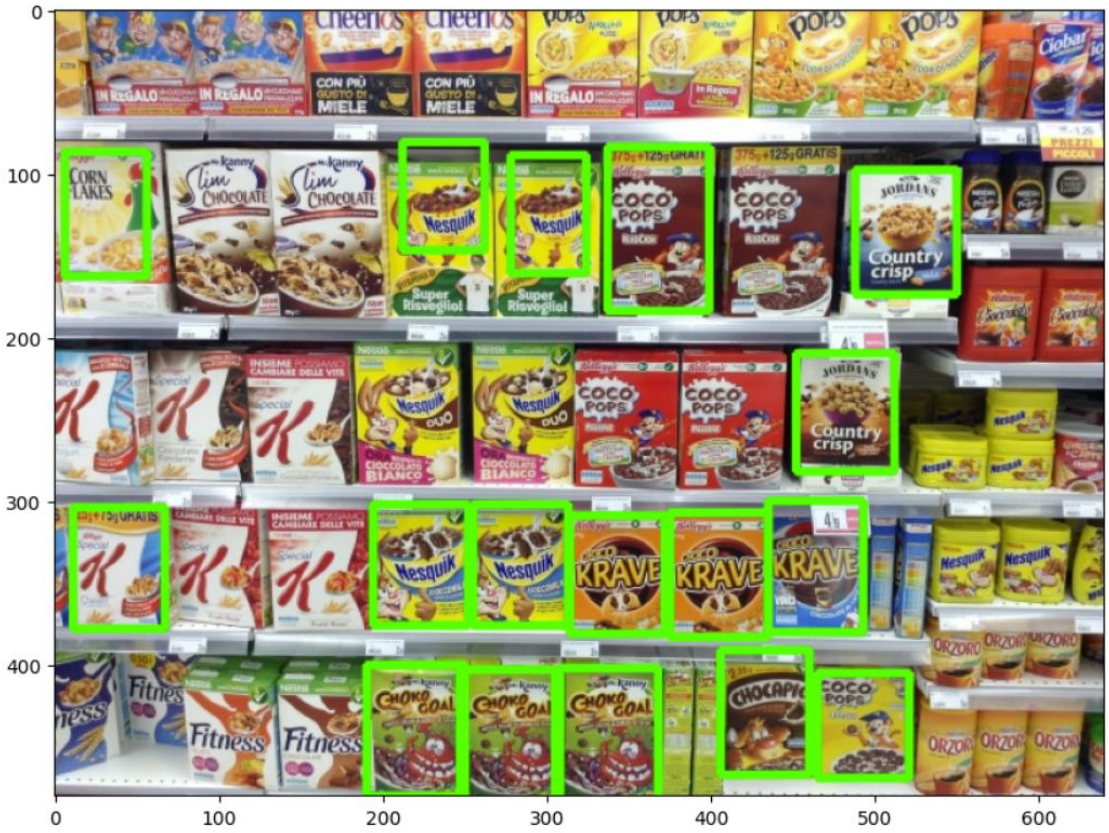
models/5.jpg - 2 instances found:

Instance 1: - position: [237, 114] - width: 50px - height: 66px

Instance 2: - position: [301, 125] - width: 47px - height: 72px

models/7.jpg - 1 instances found:

```
          Instance 1: - position: [368, 134] - width: 62px - height: 101px
models/9.jpg - 1 instances found:
          Instance 1: - position: [39, 341] - width: 56px - height: 75px
models/11.jpg - 2 instances found:
          Instance 1: - position: [405, 345] - width: 60px - height: 77px
          Instance 2: - position: [343, 345] - width: 59px - height: 73px
models/14.jpg - 1 instances found:
          Instance 1: - position: [433, 429] - width: 55px - height: 77px
models/15.jpg - 1 instances found:
          Instance 1: - position: [493, 437] - width: 56px - height: 64px
models/18.jpg - 1 instances found:
          Instance 1: - position: [482, 246] - width: 61px - height: 73px
models/19.jpg - 1 instances found:
          Instance 1: - position: [519, 136] - width: 62px - height: 76px
```



Result for scenes/h5.jpg