Detect multiple objects

Note: Cloud Vision now supports offline **asynchronous batch image annotation** for all features. This asynchronous request supports up to 2000 image files and returns response JSON files that are stored in your Google Cloud Storage bucket. For more information about this feature, refer to Offline batch image annotation (https://cloud.google.com/vision/docs/batch).

The Cloud Vision API can detect and extract multiple objects in an image with **Object Localization**.

Object localization identifies multiple objects in an image and provides a LocalizedObjectAnnotation (https://cloud.google.com/vision/docs/reference/rest/v1p3beta1/images/annotate#LocalizedObjectAnnotation) for each object in the image. Each `LocalizedObjectAnnotation` identifies information about the object, the position of the object, and rectangular bounds for the region of the image that contains the object.

Object localization identifies both significant and less-prominent objects in an image.

Object information is returned in English only. The Cloud Translation (https://cloud.google.com/translate/) can translate English labels into any of a number of other languages (https://cloud.google.com/translate/docs/languages).

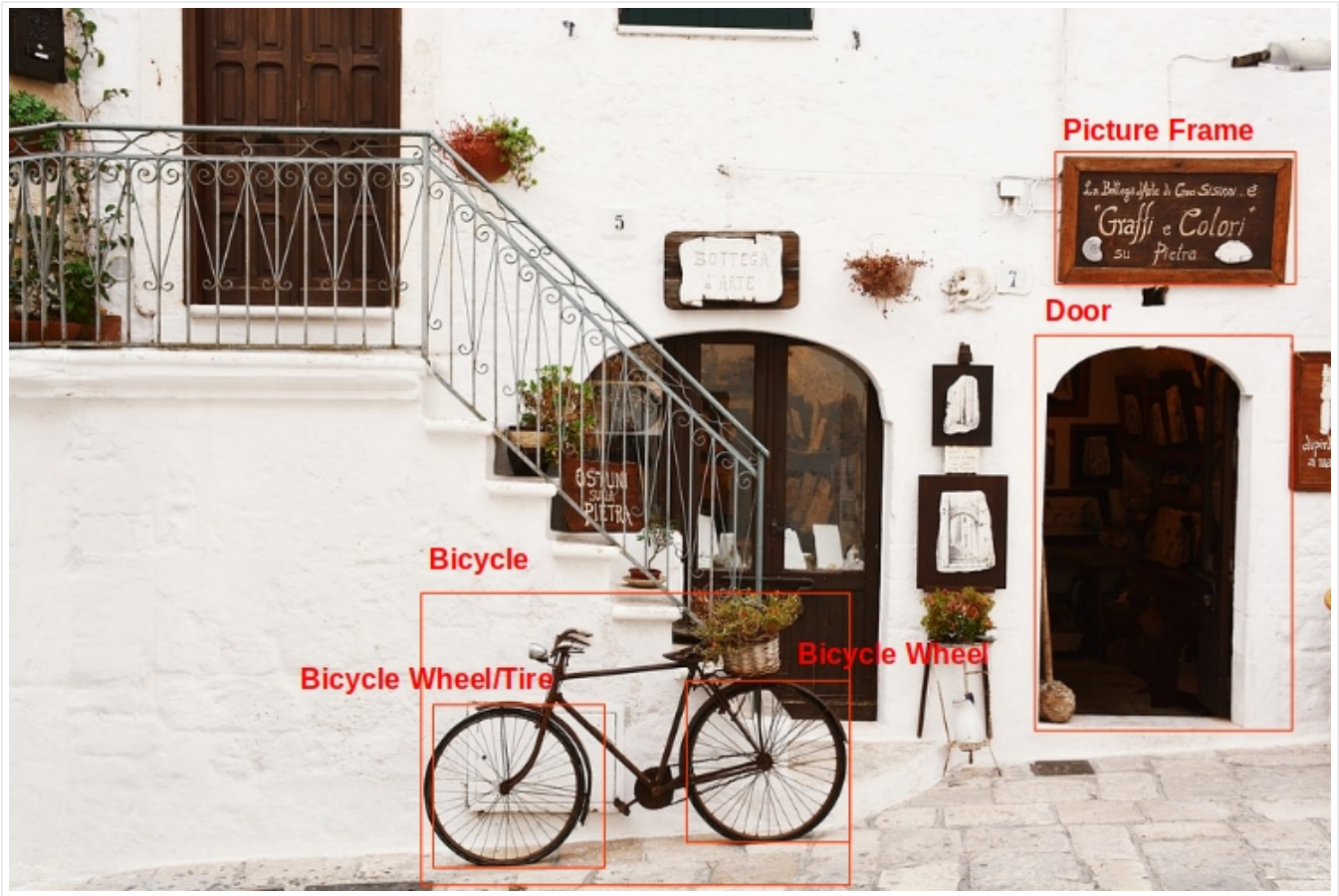

Image credit: Bogdan Dada (https://unsplash.com/photos/J9cBJjlpYKU) on Unsplash (https://unsplash.com/) (*annotations added*).

For example, the API might return the following information and bounding location data for the objects in the image above:

Name	mid	Score	Bounds
Bicycle wheel	/m/01bqk0	0.89648587	(0.32076266, 0.78941387), (0.43812272, 0.78941387), (0.43812272, 0.97331065), (0.32076266, 0.97331065)
Bicycle	/m/0199g	0.886761	(0.312, 0.6616471), (0.638353, 0.6616471), (0.638353, 0.9705882), (0.312, 0.9705882)
Bicycle wheel	/m/01bqk0	0.6345275	(0.5125398, 0.760708), (0.6256646, 0.760708), (0.6256646, 0.94601655), (0.5125398, 0.94601655)
Picture frame	/m/06z37_	0.6207608	(0.79177403, 0.16160682), (0.97047985, 0.16160682), (0.97047985, 0.31348917), (0.79177403, 0.31348917)

Name	mid	Score	Bounds
Tire	/m/0h9mv	0.55886006	(0.32076266, 0.78941387), (0.43812272, 0.78941387), (0.43812272, 0.97331065), (0.32076266, 0.97331065)
Door	/m/02dgv	0.5160098	(0.77569866, 0.37104446), (0.9412425, 0.37104446), (0.9412425, 0.81507325), (0.77569866, 0.81507325)

mid contains a machine-generated identifier (MID) corresponding to a label's Google Knowledge Graph (https://www.google.com/intl/bn/insidesearch/features/search/knowledge.html) entry. For information on inspecting **mid** values, see the Google Knowledge Graph Search API (https://developers.google.com/knowledge-graph/) documentation.

Object Localization requests

Set up your GCP project and authentication

If you have not created a Google Cloud Platform (GCP) project (https://cloud.google.com/docs/overview/#projects) and service account credentials, do so now. Expand this section for instructions.

1. Sign in (https://accounts.google.com/Login) to your Google Account.

   If you don't already have one, sign up for a new account (https://accounts.google.com/SignUp).

2. Set up a Cloud Console project.

   SET UP A PROJECT

   Click to:
   - Create or select a project.
   - Enable the Cloud Vision API for that project.
   - Create a service account.
   - Download a private key as JSON.

   You can view and manage these resources at any time in the Cloud Console (https://console.cloud.google.com/).

3. Set the environment variable ***GOOGLE_APPLICATION_CREDENTIALS*** to the file path of the JSON file that contains your service account key. This variable only applies to your current shell session, so if you open a new session, set the variable again.

Example: Linux or macOS

Replace ***[PATH]*** with the file path of the JSON file that contains your service account key.

```
export GOOGLE_APPLICATION_CREDENTIALS="[PATH]"
```

For example:

```
export GOOGLE_APPLICATION_CREDENTIALS="/home/user/Downloads/service-account-fil
```

Example: Windows

Replace ***[PATH]*** with the file path of the JSON file that contains your service account key, and ***[FILE_NAME]*** with the filename.

With PowerShell:

```
$env:GOOGLE_APPLICATION_CREDENTIALS="[PATH]"
```

For example:

```
$env:GOOGLE_APPLICATION_CREDENTIALS="C:\Users\username\Downloads\[FILE_NAME].js
```

With command prompt:

```
set GOOGLE_APPLICATION_CREDENTIALS=[PATH]
```

4. Install and initialize the Cloud SDK (https://cloud.google.com/sdk/docs/).

Detect objects in a local image

The Vision API can perform feature detection on a local image file by sending the contents of the image file as a base64 encoded (https://cloud.google.com/vision/docs/base64) string in the body of your request.

REST & CMD LINE | C# | GO | MORE

Before using any of the request data below, make the following replacements:

- ***base64-encoded-image***: The base64 representation (ASCII string) of your binary image data. This string should look similar to the following string:

- `/9j/4QAYRXhpZgAA...9tAVx/zDQDlGxn//2Q==`

Visit the base64 encode (https://cloud.google.com/vision/docs/base64) topic for more information.

HTTP method and URL:

```
POST https://vision.googleapis.com/v1/images:annotate
```

Request JSON body:

```
{
  "requests": [
    {
      "image": {
        "content": "base64-encoded-image"
      },
      "features": [
        {
          "maxResults": 10,
          "type": "OBJECT_LOCALIZATION"
        },
      ]
    }
  ]
}
```

To send your request, choose one of these options:

CURL POWERSHELL

Note: If you are not executing the command below from Cloud Shell (https://cloud.google.com/shell/docs) or Compute Engine (https://cloud.google.com/compute/docs), ensure you have set the **GOOGLE_APPLICATION_CREDENTIALS** (https://cloud.google.com/docs/authentication/production) environment variable to your service account private key file path.

Save the request body in a file called `request.json`, and execute the following command:

```
curl -X POST \
-H "Authorization: Bearer "$(gcloud auth application-default print-access-token)
-H "Content-Type: application/json; charset=utf-8" \
```

```
-d @request.json \
https://vision.googleapis.com/v1/images:annotate
```

If the request is successful, the server returns a `200 OK` HTTP status code and the response in JSON format.

Response:

Note: Zero coordinate values omitted. When the API detects a coordinate ("x" or "y") value of 0, ***that coordinate is omitted in the JSON response***. Thus, a response with a bounding poly around the entire image would be **[{},{"x": 100},{"x": 100,"y": 100},{"y": 100}] for an image that is 100px by 100px**. For more information, see the API Reference documentation (https://cloud.google.com/vision/docs/reference/rest/v1/images/annotate#boundingpoly).

Response


```
{
  "responses": [
    {
      "localizedObjectAnnotations": [
        {
          "mid": "/m/01bqk0",
          "name": "Bicycle wheel",
          "score": 0.89648587,
          "boundingPoly": {
            "normalizedVertices": [
              {
                "x": 0.32076266,
                "y": 0.78941387
              },
              {
                "x": 0.43812272,
                "y": 0.78941387
              },
              {
                "x": 0.43812272,
                "y": 0.97331065
              },
              {
                "x": 0.32076266,
```

```
            "y": 0.97331065
          }
        ]
      }
    },
    {
      "mid": "/m/0199g",
      "name": "Bicycle",
      "score": 0.886761,
      "boundingPoly": {
        "normalizedVertices": [
          {
            "x": 0.312,
            "y": 0.6616471
          },
          {
            "x": 0.638353,
            "y": 0.6616471
          },
          {
            "x": 0.638353,
            "y": 0.9705882
          },
          {
            "x": 0.312,
            "y": 0.9705882
          }
        ]
      }
    },
    {
      "mid": "/m/01bqk0",
      "name": "Bicycle wheel",
      "score": 0.6345275,
      "boundingPoly": {
        "normalizedVertices": [
          {
            "x": 0.5125398,
            "y": 0.760708
          },
          {
            "x": 0.6256646,
            "y": 0.760708
          },
          {
```

```
            "x": 0.6256646,
            "y": 0.94601655
          },
          {
            "x": 0.5125398,
            "y": 0.94601655
          }
        ]
      }
    },
    {
      "mid": "/m/06z37_",
      "name": "Picture frame",
      "score": 0.6207608,
      "boundingPoly": {
        "normalizedVertices": [
          {
            "x": 0.79177403,
            "y": 0.16160682
          },
          {
            "x": 0.97047985,
            "y": 0.16160682
          },
          {
            "x": 0.97047985,
            "y": 0.31348917
          },
          {
            "x": 0.79177403,
            "y": 0.31348917
          }
        ]
      }
    },
    {
      "mid": "/m/0h9mv",
      "name": "Tire",
      "score": 0.55886006,
      "boundingPoly": {
        "normalizedVertices": [
          {
            "x": 0.32076266,
            "y": 0.78941387
          },
```

```
            {
              "x": 0.43812272,
              "y": 0.78941387
            },
            {
              "x": 0.43812272,
              "y": 0.97331065
            },
            {
              "x": 0.32076266,
              "y": 0.97331065
            }
          ]
        }
      },
      {
        "mid": "/m/02dgv",
        "name": "Door",
        "score": 0.5160098,
        "boundingPoly": {
          "normalizedVertices": [
            {
              "x": 0.77569866,
              "y": 0.37104446
            },
            {
              "x": 0.9412425,
              "y": 0.37104446
            },
            {
              "x": 0.9412425,
              "y": 0.81507325
            },
            {
              "x": 0.77569866,
              "y": 0.81507325
            }
          ]
        }
      }
    ]
  }
]
}
```

Detect objects in a remote image

For your convenience, the Vision API can perform feature detection directly on an image file located in Google Cloud Storage or on the Web without the need to send the contents of the image file in the body of your request.

Caution: When fetching images from HTTP/HTTPS URLs, Google cannot guarantee that the request will be completed. Your request may fail if the specified host denies the request (for example, due to request throttling or DOS (https://en.wikipedia.org/wiki/Denial-of-service_attack) prevention), or if Google throttles requests to the site for abuse prevention. **You should not depend on externally-hosted images for production applications**.

REST & CMD LINE C# MORE

Before using any of the request data below, make the following replacements:

- ***cloud-storage-image-uri***: the path to a valid image file in a Cloud Storage bucket. You must at least have read privileges to the file. Example:
  - `https://cloud.google.com/vision/docs/images/bicycle_example.png`

HTTP method and URL:

```
POST https://vision.googleapis.com/v1/images:annotate
```

Request JSON body:

```
{
  "requests": [
    {
      "image": {
        "source": {
          "imageUri": "cloud-storage-image-uri"
        }
      },
      "features": [
        {
          "maxResults": 10,
          "type": "OBJECT_LOCALIZATION"
```

```
        },
      ]
    }
  ]
}
```

To send your request, choose one of these options:

CURL POWERSHELL

Note: If you are not executing the command below from Cloud Shell (https://cloud.google.com/shell/docs) or Compute Engine (https://cloud.google.com/compute/docs), ensure you have set the **`GOOGLE_APPLICATION_CREDENTIALS`** (https://cloud.google.com/docs/authentication/production) environment variable to your service account private key file path.

Save the request body in a file called `request.json`, and execute the following command:

```
curl -X POST \
-H "Authorization: Bearer "$(gcloud auth application-default print-access-token)
-H "Content-Type: application/json; charset=utf-8" \
-d @request.json \
https://vision.googleapis.com/v1/images:annotate
```

If the request is successful, the server returns a `200 OK` HTTP status code and the response in JSON format.

Response:

Note: Zero coordinate values omitted. When the API detects a coordinate ("x" or "y") value of 0, ***that coordinate is omitted in the JSON response***. Thus, a response with a bounding poly around the entire image would be
[{},{"x": 100},{"x": 100,"y": 100},{"y": 100}] for an image that is 100px by 100px. For more information, see the API Reference documentation (https://cloud.google.com/vision/docs/reference/rest/v1/images/annotate#boundingpoly).

Response

```
{
  "responses": [
    {
      "localizedObjectAnnotations": [
        {
          "mid": "/m/01bqk0",
          "name": "Bicycle wheel",
          "score": 0.89648587,
          "boundingPoly": {
            "normalizedVertices": [
              {
                "x": 0.32076266,
                "y": 0.78941387
              },
              {
                "x": 0.43812272,
                "y": 0.78941387
              },
              {
                "x": 0.43812272,
                "y": 0.97331065
              },
              {
                "x": 0.32076266,
                "y": 0.97331065
              }
            ]
          }
        },
        {
          "mid": "/m/0199g",
          "name": "Bicycle",
          "score": 0.886761,
          "boundingPoly": {
            "normalizedVertices": [
              {
                "x": 0.312,
                "y": 0.6616471
              },
              {
                "x": 0.638353,
                "y": 0.6616471
              },
              {
                "x": 0.638353,
```

```
            "y": 0.9705882
          },
          {
            "x": 0.312,
            "y": 0.9705882
          }
        ]
      }
    },
    {
      "mid": "/m/01bqk0",
      "name": "Bicycle wheel",
      "score": 0.6345275,
      "boundingPoly": {
        "normalizedVertices": [
          {
            "x": 0.5125398,
            "y": 0.760708
          },
          {
            "x": 0.6256646,
            "y": 0.760708
          },
          {
            "x": 0.6256646,
            "y": 0.94601655
          },
          {
            "x": 0.5125398,
            "y": 0.94601655
          }
        ]
      }
    },
    {
      "mid": "/m/06z37_",
      "name": "Picture frame",
      "score": 0.6207608,
      "boundingPoly": {
        "normalizedVertices": [
          {
            "x": 0.79177403,
            "y": 0.16160682
          },
          {
```

```
              "x": 0.97047985,
              "y": 0.16160682
            },
            {
              "x": 0.97047985,
              "y": 0.31348917
            },
            {
              "x": 0.79177403,
              "y": 0.31348917
            }
          ]
        }
      },
      {
        "mid": "/m/0h9mv",
        "name": "Tire",
        "score": 0.55886006,
        "boundingPoly": {
          "normalizedVertices": [
            {
              "x": 0.32076266,
              "y": 0.78941387
            },
            {
              "x": 0.43812272,
              "y": 0.78941387
            },
            {
              "x": 0.43812272,
              "y": 0.97331065
            },
            {
              "x": 0.32076266,
              "y": 0.97331065
            }
          ]
        }
      },
      {
        "mid": "/m/02dgv",
        "name": "Door",
        "score": 0.5160098,
        "boundingPoly": {
          "normalizedVertices": [
```

```
              {
                "x": 0.77569866,
                "y": 0.37104446
              },
              {
                "x": 0.9412425,
                "y": 0.37104446
              },
              {
                "x": 0.9412425,
                "y": 0.81507325
              },
              {
                "x": 0.77569866,
                "y": 0.81507325
              }
            ]
          }
        }
      ]
    }
  ]
}
```


Try it

Try object detection and localization below. You can use the image specified already (`https://cloud.google.com/vision/docs/images/bicycle_example.png`) or specify your own image in its place. Send the request by selecting **Execute**.

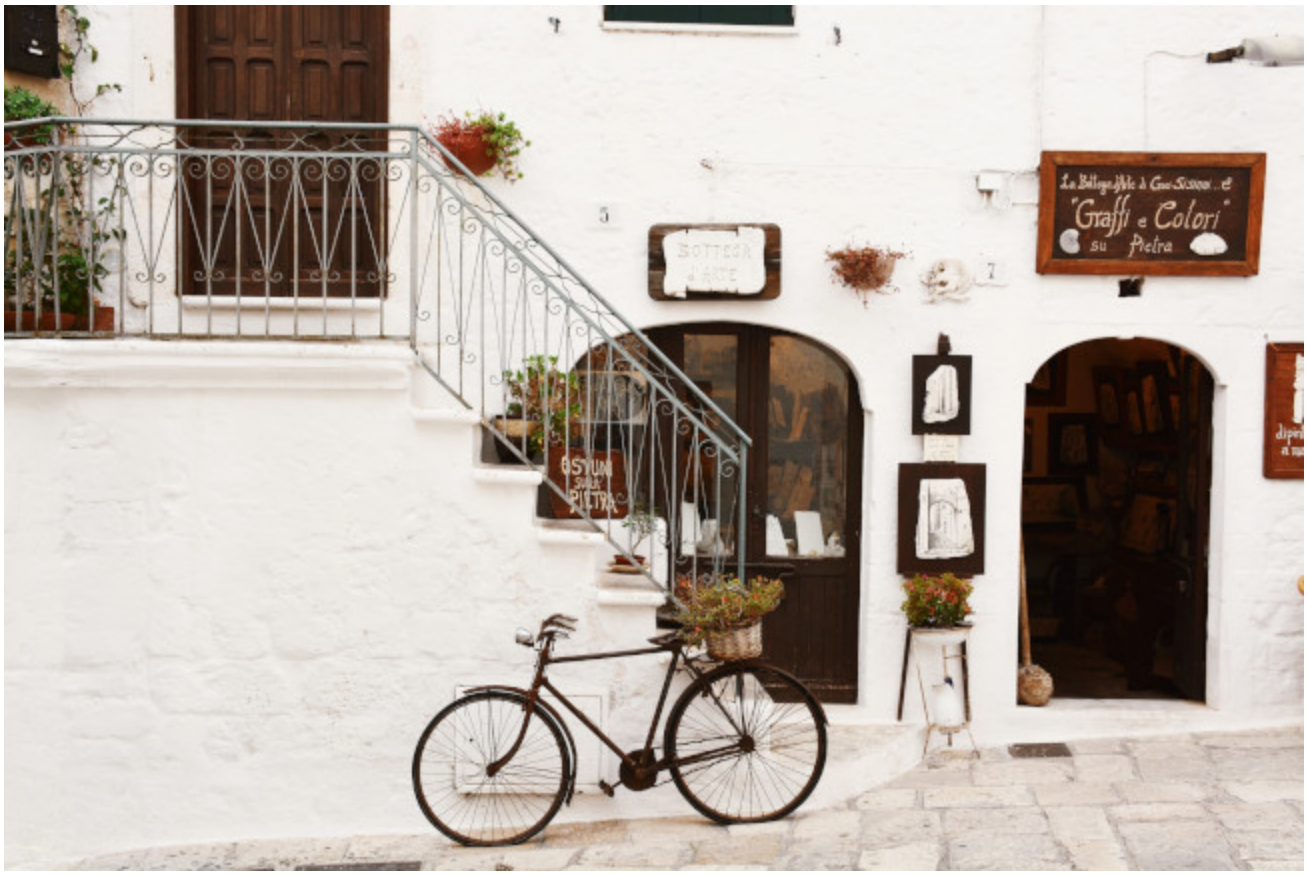

Image credit: Bogdan Dada (https://unsplash.com/photos/J9cBJjlpYKU) on Unsplash (https://unsplash.com/).

Except as otherwise noted, the content of this page is licensed under the Creative Commons Attribution 4.0 License (https://creativecommons.org/licenses/by/4.0/), and code samples are licensed under the Apache 2.0 License (https://www.apache.org/licenses/LICENSE-2.0). For details, see our Site Policies (https://developers.google.com/terms/site-policies). Java is a registered trademark of Oracle and/or its affiliates.

Last updated January 6, 2020.