

THE INTELLIGENT IMAGE

LARGE SCALE OBJECT RECOGNITION

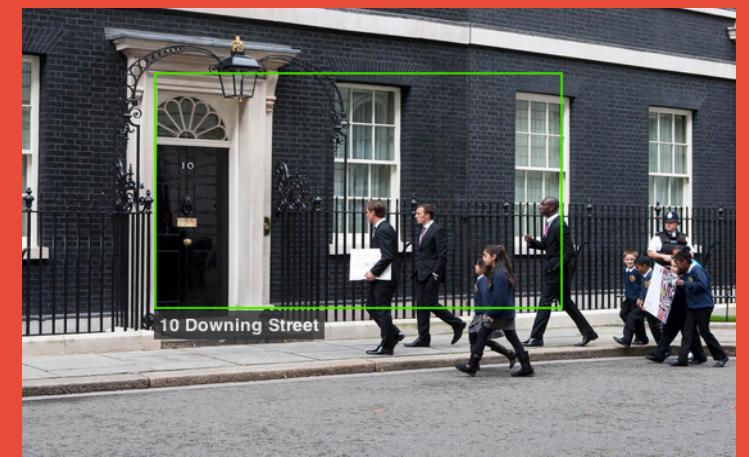
THE GOAL:
AUTOMATICALLY
RECOGNISE AND TAG
OBJECTS IN IMAGES.

IMPLEMENTATION:
A WEBSITE WHERE
USERS CAN UPLOAD A
PHOTO TO BE MADE INTO AN INTELLIGENT
IMAGE.

The screenshot shows the Wikipedia article for "Tower Bridge". The page includes a header with "Log in / create account", a navigation bar with "Article", "Talk", "Read", "Edit", "View history", and a search bar. The main content starts with a summary: "Tower Bridge (built 1886–1894) is a combined bascule and suspension bridge in London, England, over the River Thames. It is close to the Tower of London, from which it takes its name.^[1] It has become an iconic symbol of London." Below the summary is a thumbnail image of the bridge at dusk. The page lists categories like "History", "Background", "Construction", "Opening", "Design", "Hydraulic system", "Navigation control", "Reconstruction", "Misaken identity", "Tower Gateway", "London Underground station", and "Exhibition and tower walkways". It also includes sections on "Maintenance", "Design", "Dimensions", "Opening", and "Heritage status". A sidebar on the left contains links for "Main page", "Content", "Featured content", "Current events", "Random article", "Interaction", "Help", "About Wikipedia", "Community portal", "Recent changes", "Contact us", "Toolbox", "Print/export", "Languages", and "Barn-again-gü".

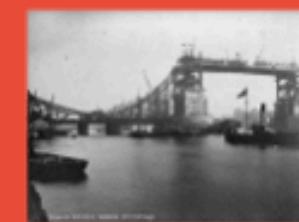


NORMAL IMAGE



INTELLIGENT IMAGE

EACH ARTICLE ON WIKIPEDIA
DEFINES AN *OBJECT*. ALL THE
IMAGES FROM AN OBJECT'S
ARTICLE ARE DOWNLOADED TO
A DATABASE AND DEFINE THE
MODEL OF THAT OBJECT



HOW IT WORKS:

1. EXTRACT VISUAL WORDS

SCALI **I**NVARIANT **F**EATURE **T**RANSFORM

$$\begin{pmatrix} x_1 & \dots & x_N \\ y_1 & \dots & y_N \\ s_1 & \dots & s_N \\ \theta_1 & \dots & \theta_N \end{pmatrix}_{4 \times N}$$

$(v_1 | \dots | v_N)_{128 \times N}$

The features of an image are detected and described by a 128-dimensional vector. The feature space is then quantized, creating “visual words”.

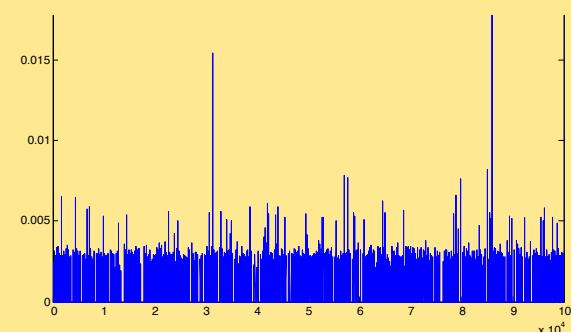
FEATURES



WORDS



HISTOGRAM



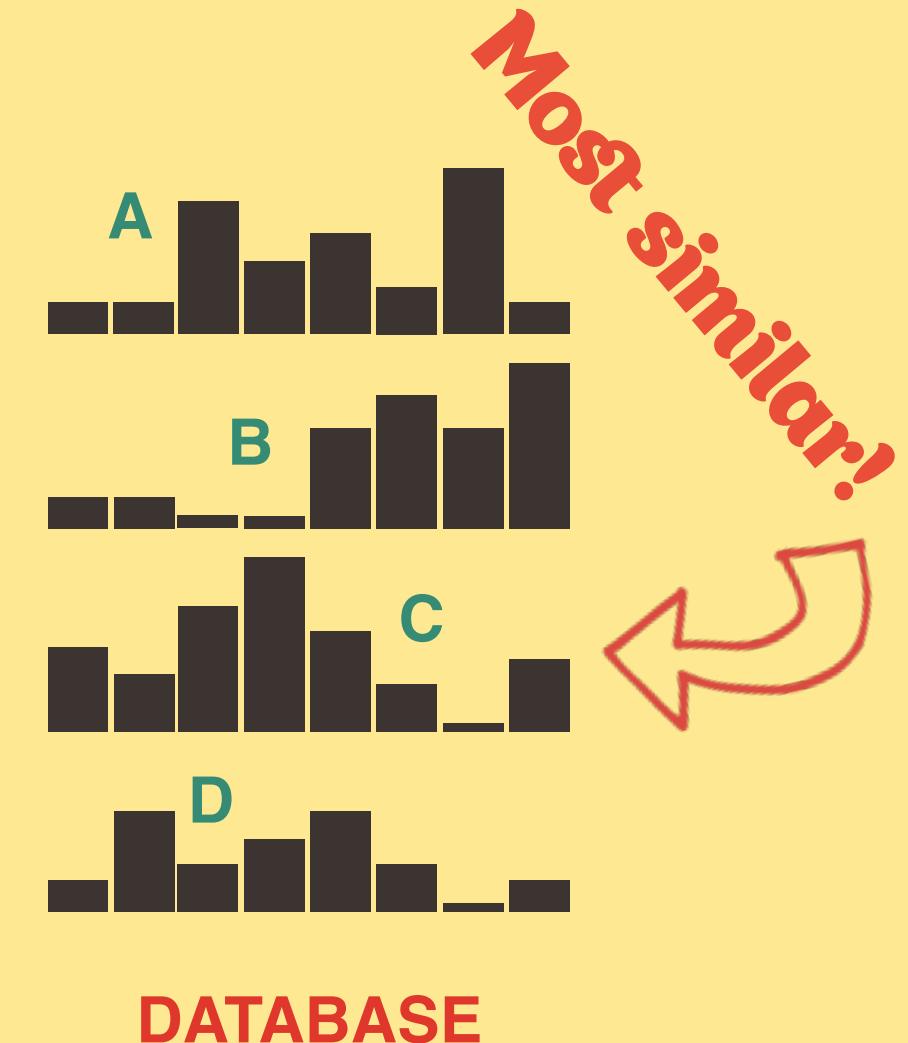
2. GOOGLE STYLE SEARCH

matches = search(Database for Words);

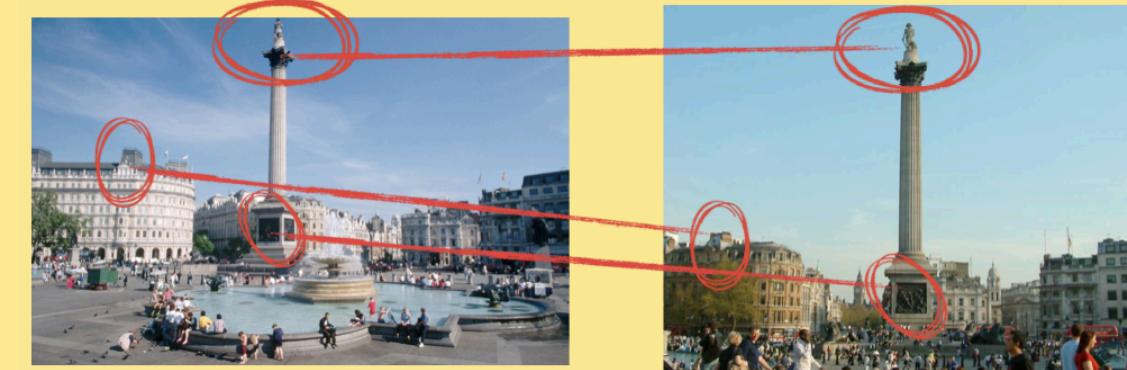
The tf-idf weighted histograms are then used to search the database for similar histograms. This is the same method as Google uses for text search, replacing words with “visual words”.



QUERY



3. SPATIALLY VERIFY



$$X_{db} = \begin{bmatrix} A & \vec{t} \\ 0 & 1 \end{bmatrix} X_{query}$$

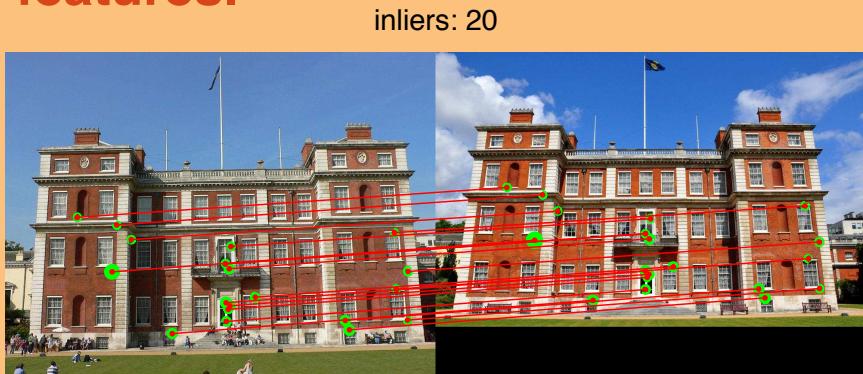
The match from the database and query image must depict the same object, so there should be a spatial correspondence between the visual words of the two images.

IMPROVEMENTS:

RANSAC



Initially RANSAC was used to estimate the affine transformation relating the visual words of two features.



This was replaced by a method called NOSAC which uses the scale of each visual word correspondence to improve the estimation of the transformation.



NOSAC

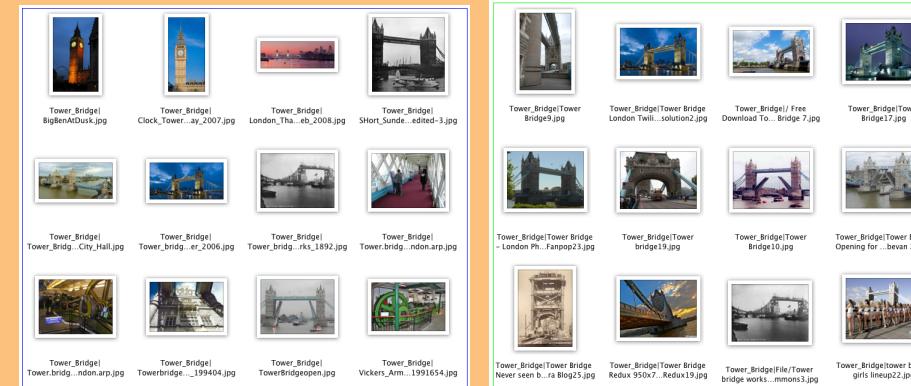
AFFINE INVARIANT

An affine invariant feature detector and descriptor was used. This means that features that are skewed due to changes in viewpoint can still be matched together. Each feature is now represented by an ellipse, with a pair of features being able to estimate a full affine transformation.

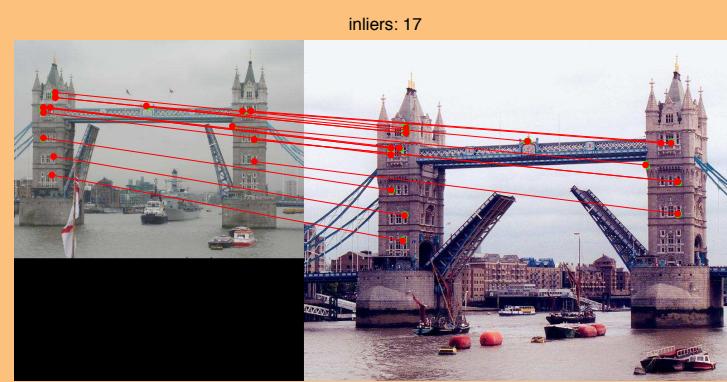
TURBO-BOOSTING

A novel method was developed to increase the information content of the database of Wikipedia images (called *model images*) by augmenting them with the visual words of images from Microsoft Bing (called *turbo images*).

1. DOWNLOAD IMAGES FROM BING FOR EACH OBJECT

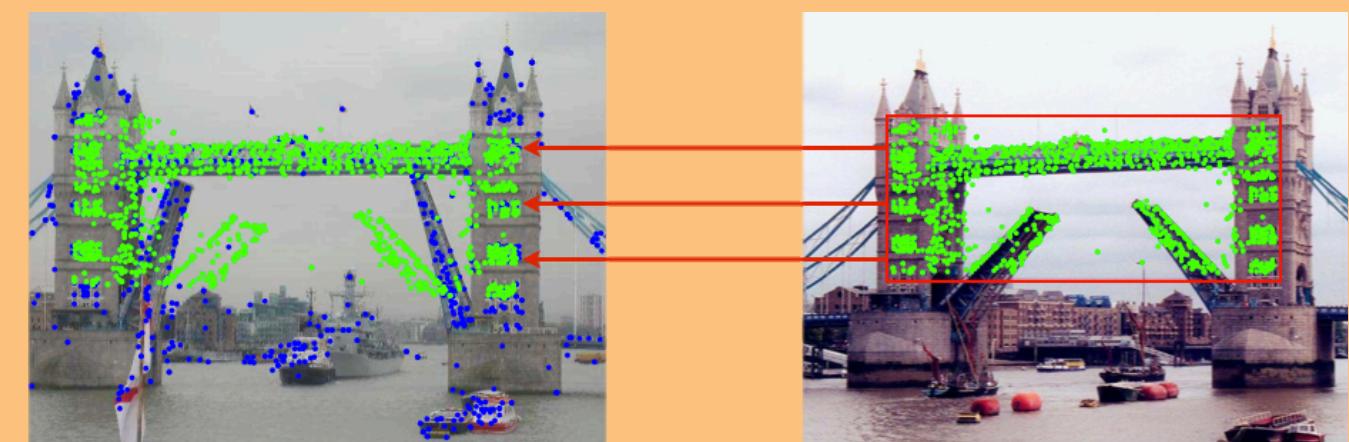


2. COMPARE EACH MODEL IMAGE WITH EACH TURBO IMAGE

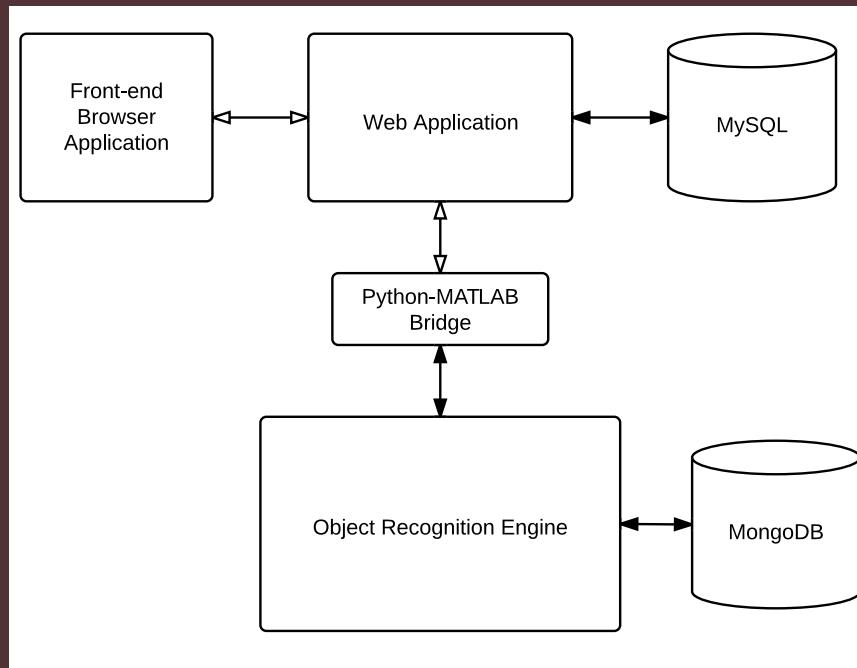


(e) The turbo image is spatially verified with the model image

3. IF A MATCH, PROJECT THE WORDS FROM THE TURBO IMAGE ON TO THE MODEL IMAGE



IMPLEMENTATION RESULTS



THE OBJECT RECOGNITION AND TAGGING ENGINE WAS BUILT IN MATLAB. THIS IS CONNECTED TO A WEB SERVER WRITTEN IN PYTHON VIA A MATLAB-PYTHON BRIDGE. THE WEB SERVER DELIVERS AN HTML AND JAVASCRIPT APP TO THE BROWSER.



INTELLIGENT IMAGE

MAX JADERBERG

Upload an image...

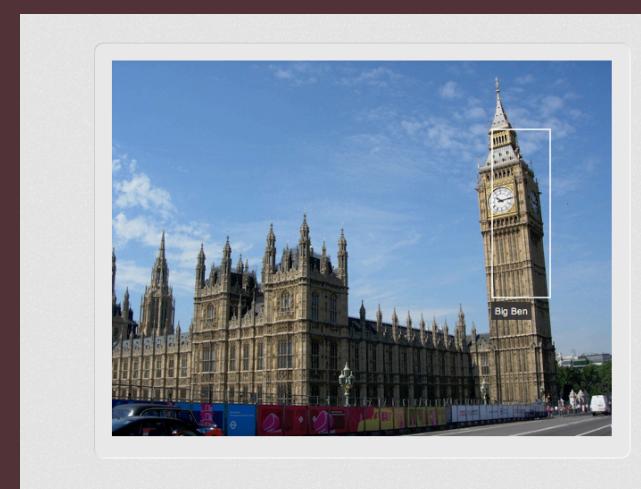
Choose File No file chosen

TAG IMAGE

TAG IMAGE

```

> Connecting to query engine...
> Querying image
> Starting match process...
> Match has a score of 1.000526e+01. Found new object, Big_Ben
> Added to matches!
> Looking for another object...
> Match has a score of 5.010743e+00. Score not large enough to be certain - no match
> Query complete
  
```



BASELINE SYSTEM:	18.1%
+ NOSAC:	20.8%
+ AFFINE INVARIANT:	22.5%
+ ROOTSIFT:	24.5%
+ TURBO-BOOSTING:	31.7%

Performance measured by *yield* - the percentage of test images successfully recognized.