ARTIFICIAL INTELLIGENCE

Final Review

Project

On

“***Image based Search Engine***”

Course code: CSE 3013

Slot: B1 + TB1

Submitted to: Prof. Annapurna Jonnalagadda

Team members:

Amey Lokhande (15BCE0926)

Anubhav Shreshta (15BCE0817)

Vineet Nalawade (15BCE0284)

CONTENTS

CHAPTER 1 PROBLEM STATEMENT

Sometimes, we need to categorize our images according to some criteria like scenarios, background, automobiles and many other different features. This can be done by using meta information for the JPEG image. But sometimes the meta-information can give incorrect information such as incorrect dates, incorrect name, or some other incorrect information. Manual categorization of images is a very tedious task. Searching by meta-data is only marginally different than your standard keyword-based search engines mentioned above. Search by meta-data systems rarely examine the contents of the image itself. Instead, they rely on textual clues such as (1) manual annotations and tagging performed by humans along with (2) automated contextual hints, such as the text that appears near the image on a webpage. When a user performs a search on a search by meta-data system they provide a query, just like in a traditional text search engine, and then images that have similar tags or annotations are returned.

CHAPTER 2 INTRODUCTION

Image search engines that quantify the contents of an image are called **Content-Based Image Retrieval** (CBIR) systems. The term CBIR is commonly used in the academic literature, but in reality, it’s simply a fancier way of saying “image search engine”, with the added poignancy that the search engine is relying *strictly* on the contents of the image and not any textual annotations associated with the image. Again, it’s important to reinforce the point that Search by Example systems rely *strictly* on the contents of the image. These types of systems tend to be extremely hard to build and scale, but allow for a fully automated algorithm to govern the search — no human intervention is required.

There tend to be three types of image search engines: **search by meta-data**, **search by example**, and a **hybrid approach** of the two.

- **Searching by meta-data** is only marginally different than our standard keyword-based search engines. Search by meta-data systems rarely examine the contents of the image itself. Instead, they rely on textual clues such as (1) manual annotations and tagging performed by humans along with (2) automated contextual hints, such as the text that appears near the image on a webpage. When a user performs a search on a search by metadata system they provide a query, just like in a traditional text search engine, and then images that have similar tags or annotations are returned. Again, when utilizing a search by meta-data system the actual image itself is rarely examined.

- **Search by example systems**, on the other hand, rely solely on the contents of the image — no keywords are assumed to be provided. The image is analyzed, quantified, and stored so that similar images are returned by the system during a search. Instead of manually labelling all the images in the database, we utilize some sort of algorithm to extract “features” (i.e., a list of numbers to quantify and abstractly represent the image) from the image itself. Then, when a user submits a query image, we extract features from the query image and compare them to our database of features and try to find similar images.

- There is a middle ground between the two known as **hybrid approach** which correlates the features extracted from the image with the text. Using this approach, we could build an image search engine that could take both contextual hints along with a Search by Example strategy.

Motivation

Image databases and collections can be enormous in size, containing hundreds, thousands or even millions of images. In recent years, a drastic increase in the size of image databases has been realized. So, the conventional method of image retrieval i.e., searching for a keyword that would match the descriptive keyword assigned to the image by a human categorizer has become inefficient. Currently under development, even though several systems exist, is the retrieval of images based on their content, called Content Based Image Retrieval, CBIR. While computationally expensive, the results are far more accurate than conventional image indexing. Hence, there exists a tradeoff between accuracy and computational cost. This tradeoff decreases as more efficient algorithms are utilized and increased computational power becomes inexpensive.

Significance

Efficient image browsing, image retrieval and searching tools are needed to users in various domains including remote sensing, crime prevention, fashion, medicine, publishing, architecture, etc. So, we need an effective system for cataloguing, annotating, and accessing these data that have been significantly requested in various applications such as image search engine, grouping and filtering of Web images, biomedical information management, computer forensics, and security. Therefore, we need to focus our attention on image retrieval methods.

Scope

Content based image retrieval schemes proposed here are promising but still, there is significant scope for future research. This research work improves the understanding of various techniques for feature extraction and similarity measurement which aids image retrieval and advances the state-of the art through its contributions. The incorporation of various techniques presented in this research work raises a number of challenges for further research such as validate the practicality of proposed model in real environments.

Applications

Potential uses for CBIR include:

- Architectural and engineering design
- Art collections
- Crime prevention
- Geographical information and remote sensing systems
- Intellectual property
- Medical diagnosis
- Military
- Photograph archives
- Retail catalogs
- Nudity-detection filters
- Face Finding

CHAPTER 3 LITERATURE SURVEY

 Related work so far happened:

Query by image content (QBIC) also referred as content based image retrieval (CBIR). QBIC performs the image retrieval by passing the content of query image like shape, color, texture, spatial location. There are various techniques available to extract the image content and to perform the image retrieval. CBIR has its application in various fields like digital libraries, medical field, military, facial recognition to prevent crime etc. An image has three types of contents as texture, color, shape which is used in content based image retrieval. Among all the three features for color image retrieval, color feature is used. It is not variable to complex background and independent of image orientation and size.

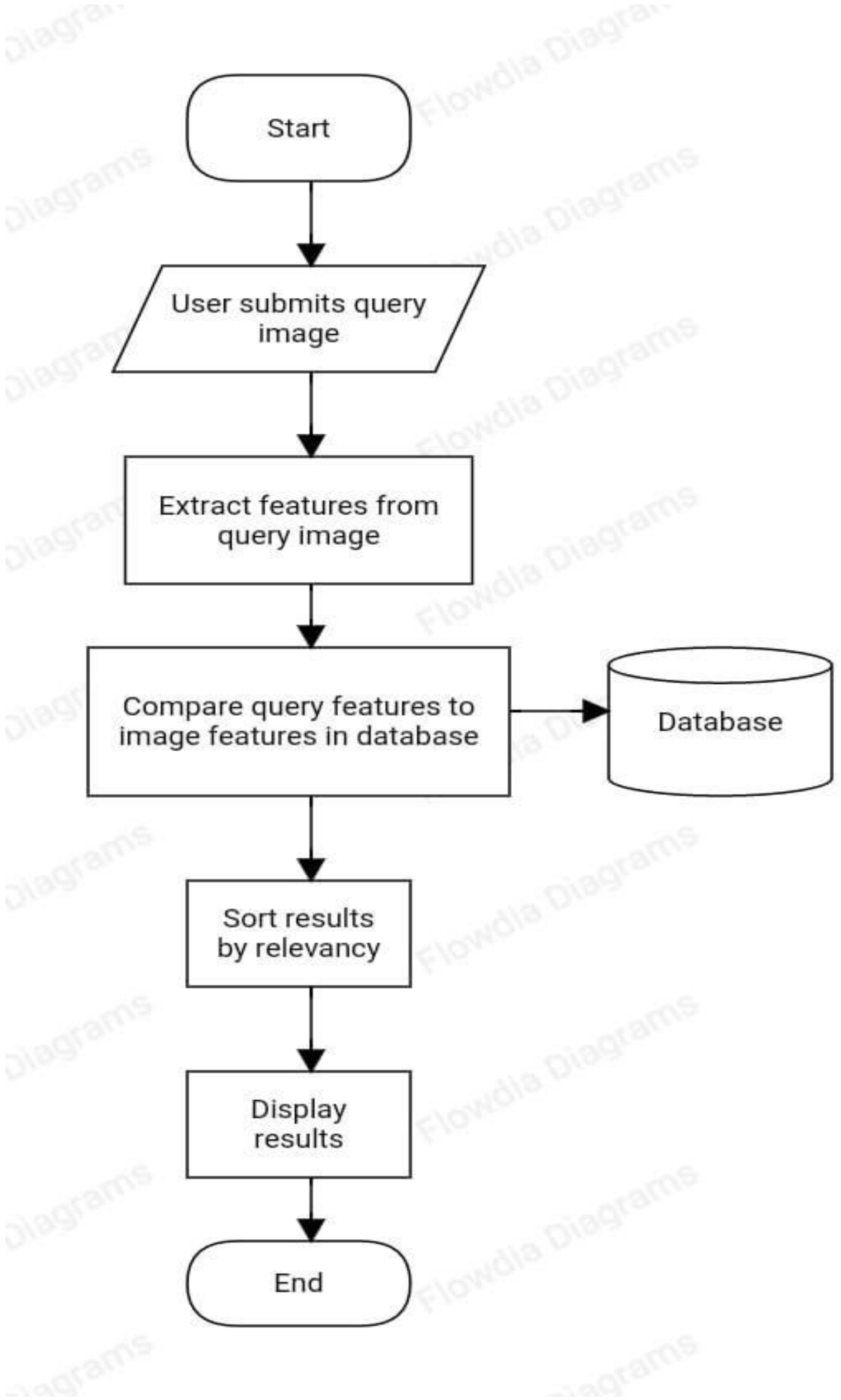
Color is invariant to complexity and very much sensitive to humans than the grayscale images. There are various techniques available to extract the color from images. Color coherence vector, color histogram, color correlogram are the main techniques which are used to extract the color feature. Basically color histogram finds the color distribution in image. When two different images have the same histogram then histogram techniques gets fail. Color Moment extracts these color distribution efficiently. Color distribution in an image is characterized by color moment so we have to find out the three order color moment- first order (mean), second order (standard deviation), third order (skewness). Further techniques which used in extracting the color feature are color coherence vector and color correlogram. Color Moment is much better then all color techniques.

Various color models are also available to perform retrieval which are RGB, HSV, L*a*b*, CMY etc. The RGB color space is well supported on image processing programs. Main disadvantage of RGB is that the RGB color space is not uniform and all three components (R, G, B) have equal importance and so those values have to be quantized with the same precision. So over RGB color space, HSV is preferred. The L*a*b* (Brightness, red-green and yellowblue content) system gives quantitative expression to the Munsell system of color classification. Now L*a*b* color space is mostly used.

Texture is another one widely used feature in content based image retrieval. Various methods are available which are used to describe texture feature such as gray-level co-occurrence matrix, Ranklet texture feature, Haar discrete wavelet transform and Gabor filter texture feature etc. Shape is also a feature of image which is used in feature extraction. Various methods to extract shape feature are edge histogram descriptor (EHD), Sobel descriptor, SIFT, Fourier descriptor of PFT etc. After feature extraction, similarity matching is mainly performed using Euclidian distance, Manhattan distance, histogram matching etc.

CHAPTER 4 IMPLEMENTATION

- **FLOWCHART OF THE PROPOSED SOLUTION:**


Start
User submits query image
Extract features from query image
Compare query features to image features in database
Database
Sort results by relevancy
Display results
End

- **IMPLEMENTATION DETAILS:**

The implementation is mainly divided into four parts:

1. **Defining your image descriptor:** At this phase we decide what aspect of the image we want to describe. Are we interested in the color of the image? The shape of an object in the image? Or do we want to characterize texture?

Instead of using a standard color histogram, we are going to apply a few tricks and make it a little more robust and powerful.

Our image descriptor will be a 3D color histogram in the HSV color space (Hue, Saturation, Value). Typically, images are represented as a 3-tuple of Red, Green, and Blue (RGB). However, while RGB values are simple to understand, the RGB color space fails to mimic how humans perceive color. Instead, we are going to use the HSV color space which maps pixel intensities into a cylinder. There are other color spaces that do an even better job at mimicking how humans perceive color, such as the CIE L*a*b* and CIE XYZ spaces, but let's keep our color model relatively simple for our first image search engine implementation.

So now that we have selected a color space, we now need to define the number of bins for our histogram. Histograms are used to give a (rough) sense of the density of pixel intensities in an image. Essentially, our histogram will estimate the probability density of the underlying function, or in this case, the probability P of a pixel color C occurring in our image I. It's important to note that there is a trade-off with the number of bins you select for your histogram. If you select too few bins, then your histogram will have less components and unable to disambiguate between images with substantially different color distributions. Likewise, if you use too many bins your histogram will have many components and images with very similar contents may be regarded and "not similar" when in reality they are.

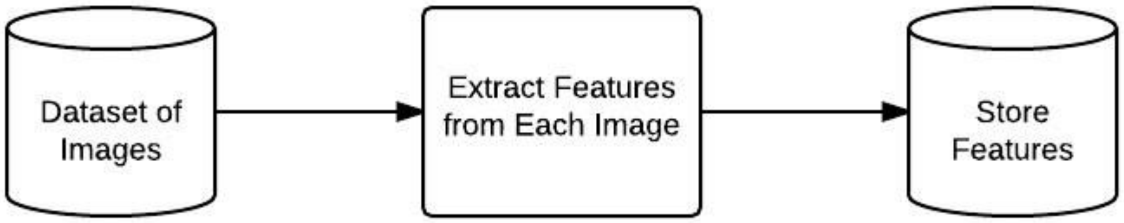
In general, you'll want to experiment with the number of bins for your color histogram descriptor as it is dependent on (1) the size of your dataset and (2) how similar the color distributions in your dataset are to each other. For our vacation photo image search engine, we'll be utilizing a 3D color histogram in the HSV color space with 8 bins for the Hue channel, 12 bins for the saturation channel, and 3 bins for the value channel, yielding a total feature vector of dimension 8 x 12 x 3 = 288.

This means that for every image in our dataset, no matter if the image is 36 x 36 pixels or 2000 x 1800 pixels, all images will be abstractly represented and quantified using only a list of 288 floating point numbers.

For our image descriptor, we are going to divide our image into five different regions: (1) the top-left corner, (2) the top-right corner, (3) the bottom-right corner, (4) the bottom-left corner, and finally (5) the center of the image.

By utilizing these regions we'll be able to mimic a crude form of localization, being able to represent our above beach image as having shades of blue sky in the top-left and top-right corners, brown sand in the bottom-left and bottom-right corners, and then a combination of blue sky and brown sand in the center region.

2. **Indexing the dataset:** Now the image descriptor is defined, the next thing is to apply image descriptor to each image in your dataset, extract features from these images, and write the features to storage (ex. CSV file, RDBMS, Redis, etc.) so that they can be later compared for similarity.



3. **Defining your similarity metric:** When we get the feature vectors. But how are you going to compare them? Popular choices include the Euclidean distance, Cosine distance, and chi-squared distance, but the actual choice is highly dependent on (1) your dataset and (2) the types of features you extracted.

We have decided to use Chi2 distance function for the similarity metric.

Our chi2_distance function requires two arguments, which are the two histograms we want to compare for similarity. An optional eps value is used to prevent division-by-zero errors.

The function gets its name from the Pearson's chi-squared test statistic which is used to compare discrete probability distributions.Since we are comparing color histograms, which are by definition probability distributions, the chi-squared function is an excellent choice.

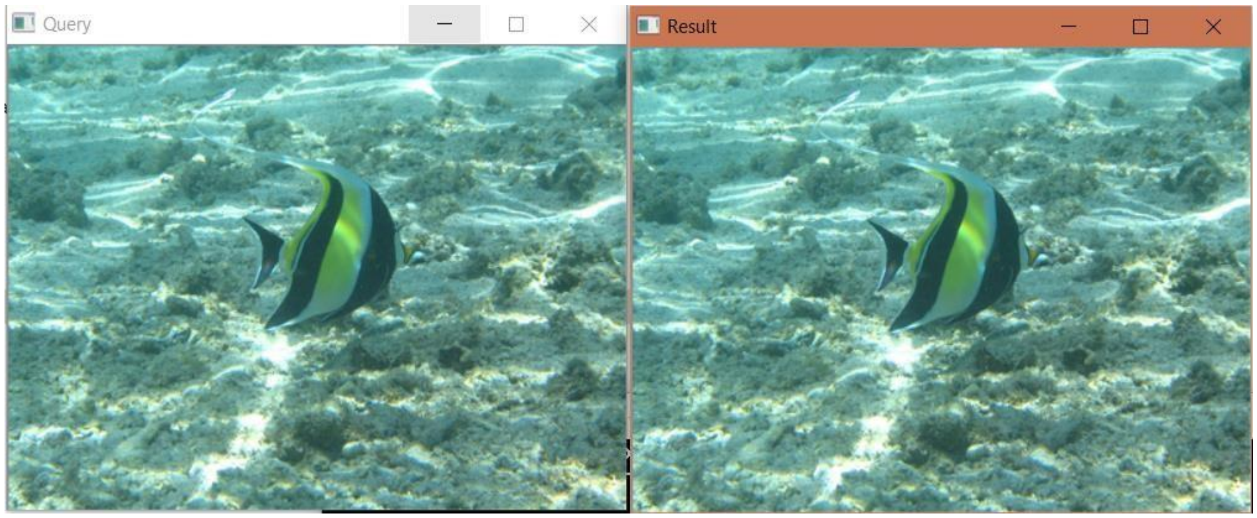
In general, the difference between large bins vs. small bins is less important and should be weighted as such — and this is exactly what the chi-squared distance function does.

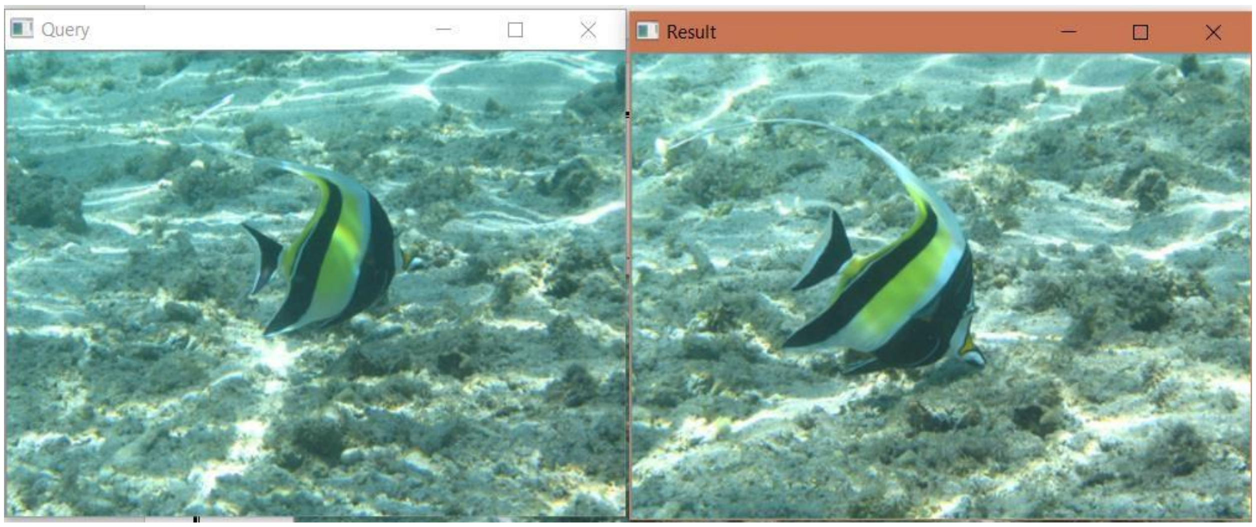
4. **Searching:** The final step is to perform an actual search. A user will submit a query image to your system (from an upload form or via a mobile app, for instance) and your job will be to (1) extract features from this query image and then (2) apply your similarity function to compare the query features to the features already indexed. From there, you simply return the most relevant results according to your similarity function.

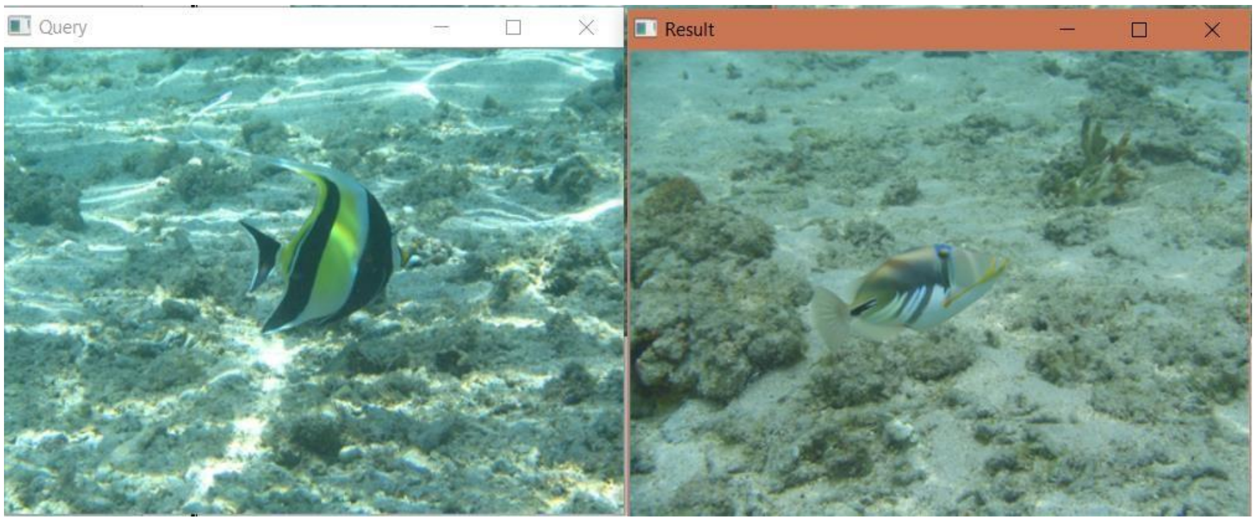
- **COMPLEXITY ANALYSIS:**

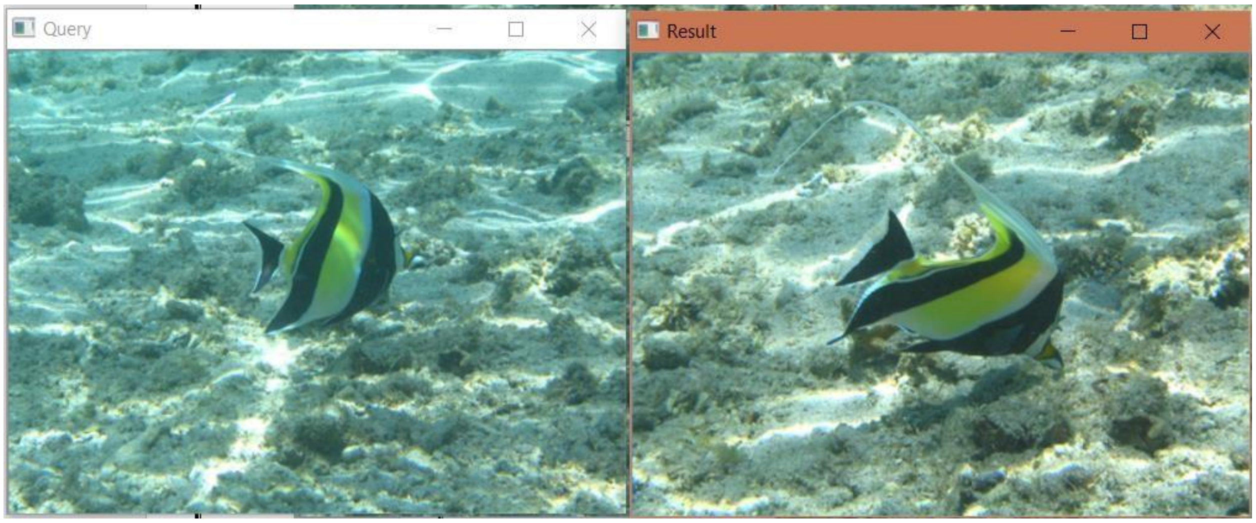
We perform a number of experiments on five real-world image datasets to evaluate the proposed methods. In the first experiment, two solving strategies, the iterative projected gradient and the soft-max method, are compared according to training time and classification error. In the second experiment, we evaluate the proposed method with the state-of-the-art methods, including four histogram metrics (, QCN (available at http://www.ariel.ac.il/sites/ofirpele/QC/), QCS (available at http://www.ariel.ac.il/sites/ofirpele/QC/), and FastEMD (available at http://www.ariel.ac.il/sites/ofirpele/FastEMD/code/)) and three metric learning methods (ITML (available at http://www.cs.utexas.edu/~pjain/itml/), LMNN (available at http://www.cse.wustl.edu/~kilian/code/files/mLMNN2.4.zip), and GB-LMNN (available at http://www.cse.wustl.edu/~kilian/code/files/mLMNN2.4.zip)), on the image retrieval dataset corel. As the source code of the closely related method -LMNN is not publicly available, we further perform the full-rank and low-rank metric learning experiments on the four datasets (dslr, webcam, amazon, and caltech). Since the -LMNN has also been tested on the above datasets, we can make a direct comparison. There are several parameters to be set in our model. The parameter is empirically set to , , 10%#NumberofTrainingSamples/#NumberofClasses. We fix the parameters and to 0.5 and 50 in our experiments, respectively. Moreover, the parameter is set to 1 if the regularization is used. In our work, all the experiments are executed in a PC with 8 Intel(R) Xeon(R) E5-1620 CPUs (3.6 GHz) and 4 GB main memory.

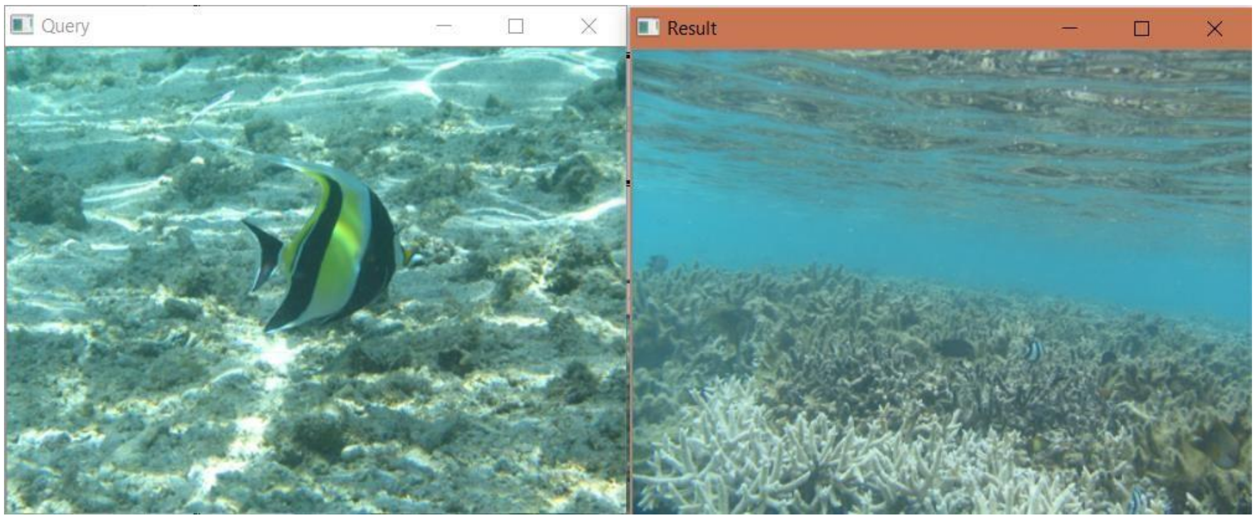
- **SCREENSHOTS OF THE PROJECT:**

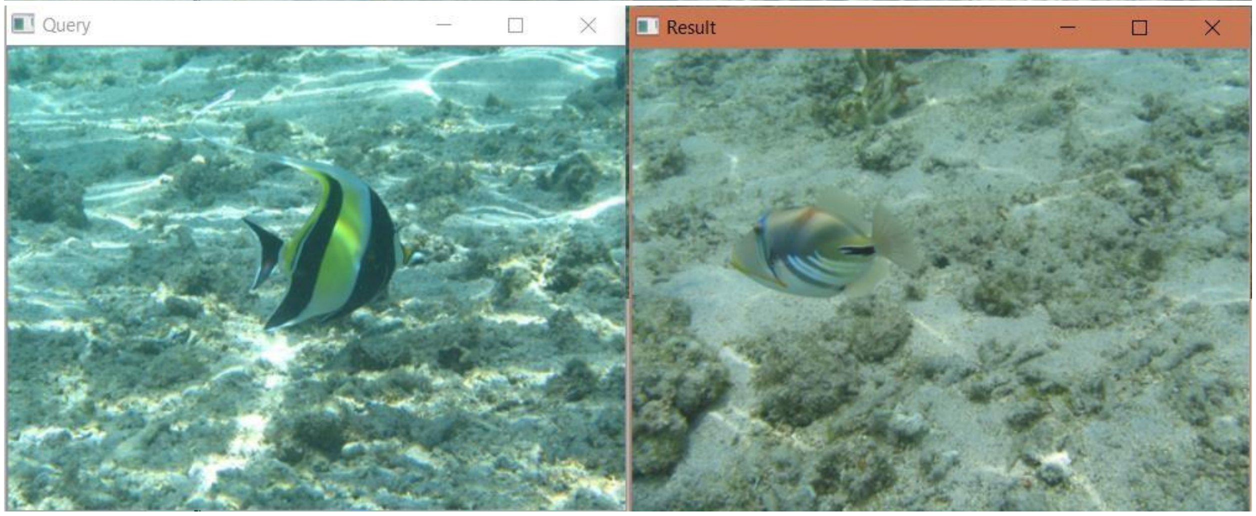


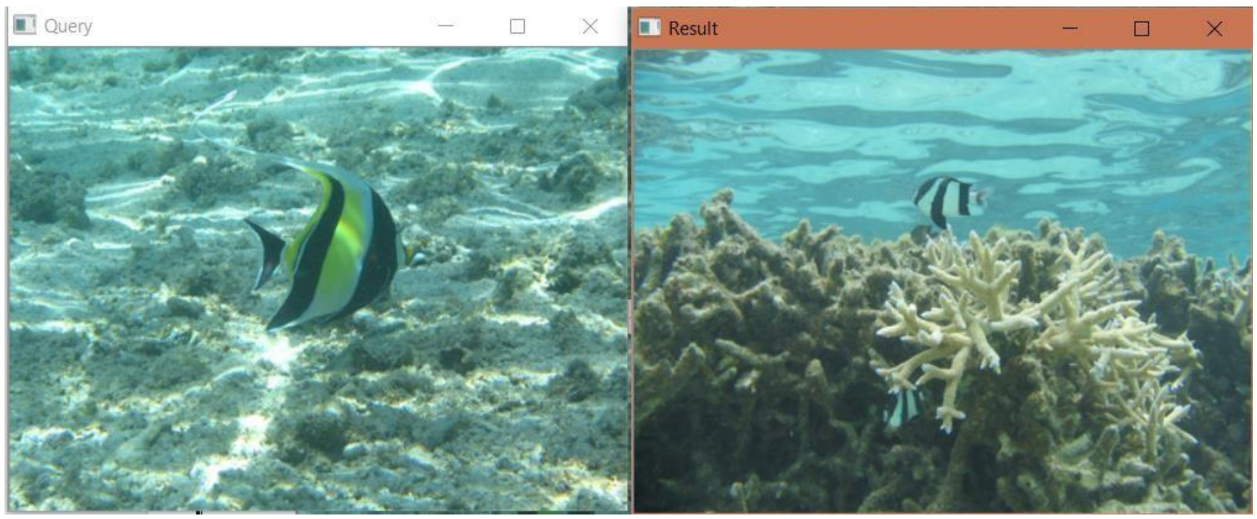

Query
Result

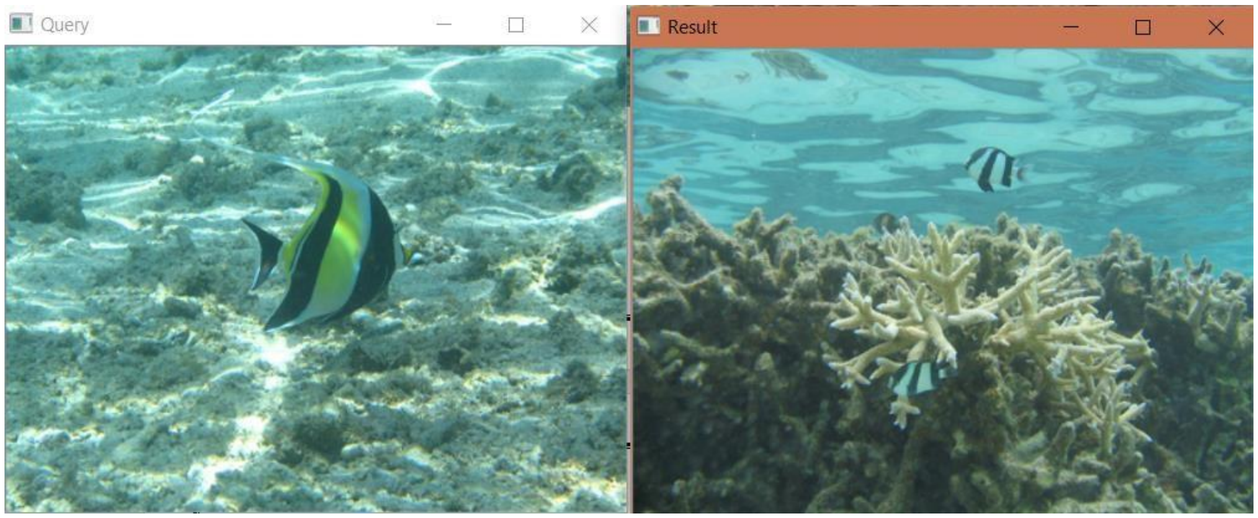

Query
Result


Query
Result


Query
Result


Query
Result


Query
Result



CHAPTER 5 FUTURE WORK

The project consisted of basic implementation of different types of content based image search engines. A few improvements and modifications that can be added to the project include,

- We can achieve better feature extraction and image processing techniques and thus optimize the accuracy and the variety of features extracted.
- query processing can be optimized to ensure improved precision and recall
- We can use multiple images at a time to ensure accurate results.

CHAPTER 6 REFERENCES

1 Yang, X., Mei, T., Zhang, Y. Liu, J., Satoh, S. (20 July 2016) Web Image Search Re-Ranking With ClickBased Similarity and Typicality. *IEEE Transactions on Image Processing*.

2 Sindhu S, Mr. C O Prakash Development of the Content Based Image Retrieval Using Color, Texture and Edge Features *International Journal of Computer Trends and Technology*.

3 Yao, B. Z., Yang, X. Lin, L. (17 June 2010). I2T: Image Parsing to Text Description. *Proceedings of the IEEE*.

4 Pradnya Vikhar, P.P. Karde and V.M. Thakare, Comprehensive Analysis of Some Recent Competitive

5 CBIR Techniques, *ICTACT JOURNAL ON IMAGE AND VIDEO PROCESSING, FEBRUARY 2017, VOLUME: 07, ISSUE: 03*.

6 Image searching on the Excite Web search engine Abby Goodrum a,*, Amanda Spink b a College of Information Science and Technology, Drexel University, 3141 Chestnut Street, Philadelphia, PA 19104-2875, USA b School of Information Sciences and Technology, The Pennsylvania State University, State College, PA 16801, USA Received 29 March 2000; accepted 14 June 2000

7 Linguistic Query Based Quality Evaluation of Selected Image Search Engines Artur Karczmarczyka *, Jarosław Jankowskia , Wojciech Sałabuna a West Pomeranian University of Technology, al. Piastów 17, 70-310 Szczecin, Poland

8 Content-Based Search Engines for construction image databases Ioannis Brilakis*, Lucio Soibelman University of Illinois at Urbana-Champaign, Department of Civil and Environmental Engineering, United States Accepted 4 November 2004

9 Text-based search engine application framework Asri Maspupah; Saiful Akbar 2016 International Conference on Data and Software Engineering (ICoDSE) Year: 2016 Pages: 1 – 6

10 Cross-media relevance mining for evaluating text-based image search engine Zhongwen Xu; Yi Yang; Ashraf Kassim; Shuicheng Yan 2014 IEEE International Conference on Multimedia and Expo Workshops (ICMEW) Year: 2014 Pages: 1 – 4 Cited by: Papers (1)