A Report for Simple OpenCv-based Image Retrieval Program

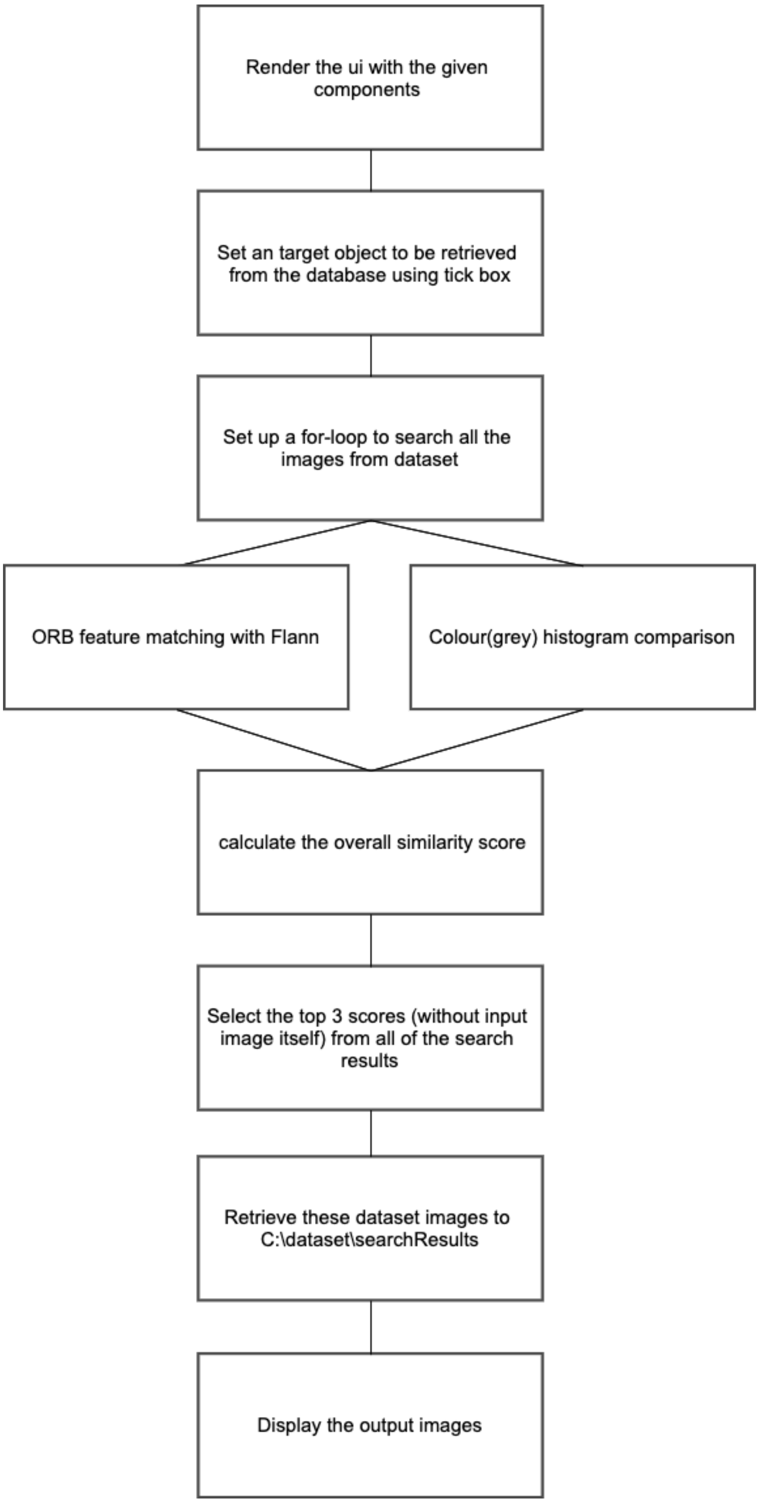
CS4185 Multimedia Technologies and Applications

Tam Chin Pang
SID: 56226481

11 November 2021

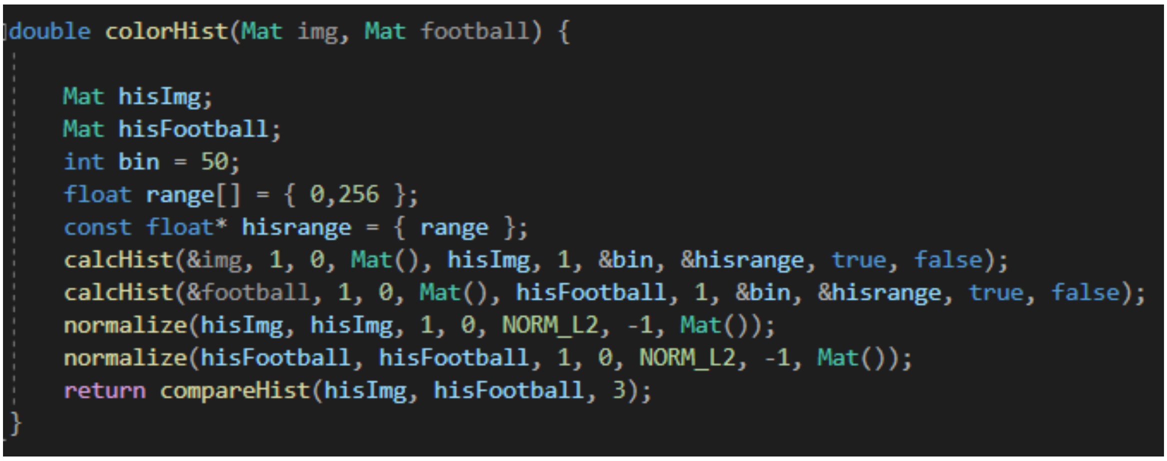
Image Retrieval

Flow of Program


Render the ui with the given components
Set an target object to be retrieved from the database using tick box
Set up a for-loop to search all the images from dataset
ORB feature matching with Flann
Colour(grey) histogram comparison
calculate the overall similarity score
Select the top 3 scores (without input image itself) from all of the search results
Retrieve these dataset images to C:\dataset\searchResults
Display the output images

Features

Single channel Histogram Comparison

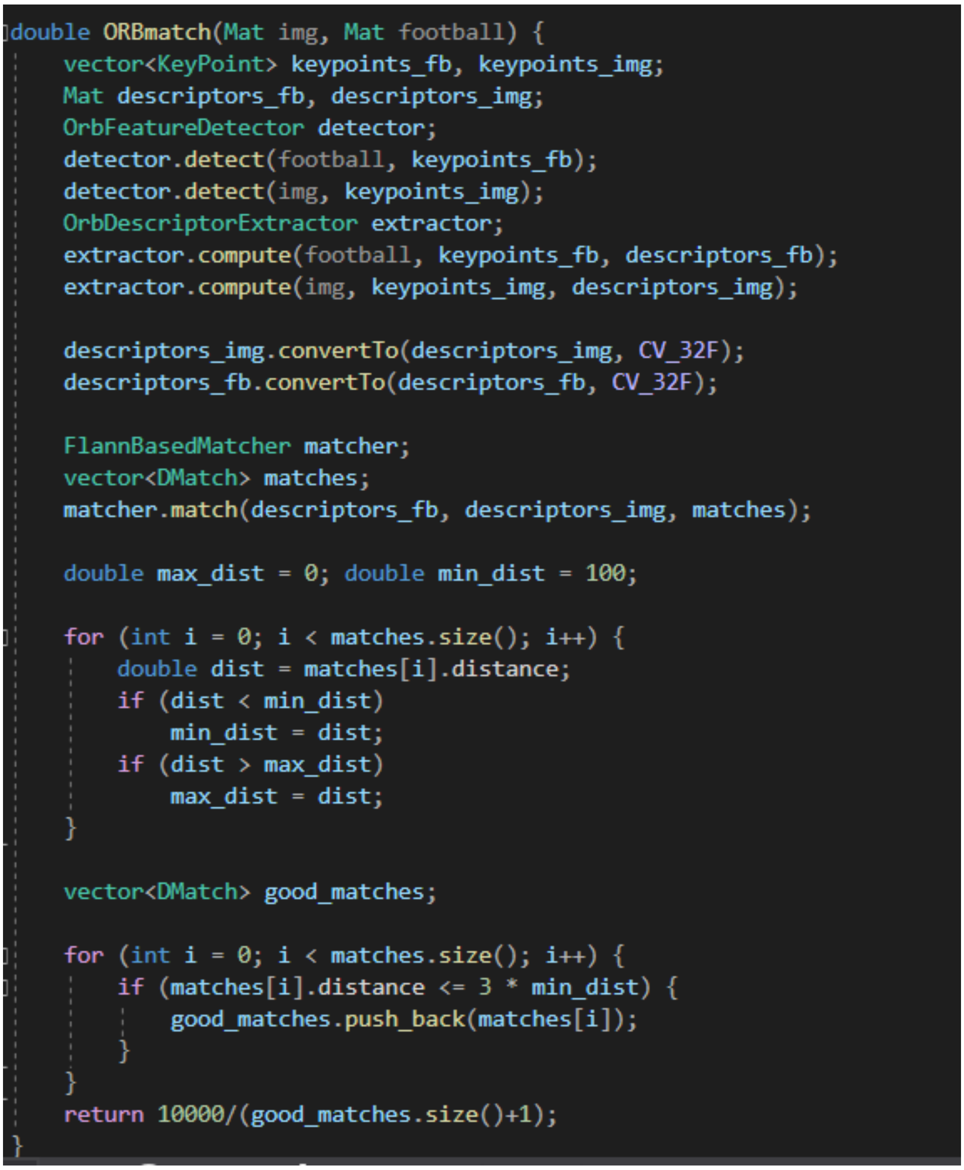


```
double colorHist(Mat img, Mat football) {

    Mat hisImg;
    Mat hisFootball;
    int bin = 50;
    float range[] = { 0,256 };
    const float* hisrange = { range };
    calcHist(&img, 1, 0, Mat(), hisImg, 1, &bin, &hisrange, true, false);
    calcHist(&football, 1, 0, Mat(), hisFootball, 1, &bin, &hisrange, true, false);
    normalize(hisImg, hisImg, 1, 0, NORM_L2, -1, Mat());
    normalize(hisFootball, hisFootball, 1, 0, NORM_L2, -1, Mat());
    return compareHist(hisImg, hisFootball, 3);
}
```

Histogram is designed the extract the colour features from the images. By comparing the colour features from two images, the image similarity score can be obtained.

The two input images are firstly converted into grey colour. Two single channel colour histograms are drawn using *calcHist()*. They are then normalised into the same scale using *normalize()*. Finally, they are compared with each other using *compareHist() to* obtain the similarity score.

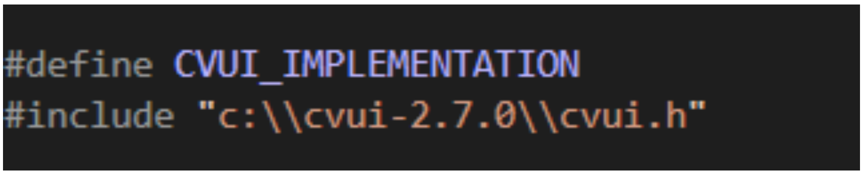
ORB features matching using Flann



```cpp
double ORBmatch(Mat img, Mat football) {
    vector<KeyPoint> keypoints_fb, keypoints_img;
    Mat descriptors_fb, descriptors_img;
    OrbFeatureDetector detector;
    detector.detect(football, keypoints_fb);
    detector.detect(img, keypoints_img);
    OrbDescriptorExtractor extractor;
    extractor.compute(football, keypoints_fb, descriptors_fb);
    extractor.compute(img, keypoints_img, descriptors_img);

    descriptors_img.convertTo(descriptors_img, CV_32F);
    descriptors_fb.convertTo(descriptors_fb, CV_32F);

    FlannBasedMatcher matcher;
    vector<DMatch> matches;
    matcher.match(descriptors_fb, descriptors_img, matches);

    double max_dist = 0; double min_dist = 100;

    for (int i = 0; i < matches.size(); i++) {
        double dist = matches[i].distance;
        if (dist < min_dist)
            min_dist = dist;
        if (dist > max_dist)
            max_dist = dist;
    }

    vector<DMatch> good_matches;

    for (int i = 0; i < matches.size(); i++) {
        if (matches[i].distance <= 3 * min_dist) {
            good_matches.push_back(matches[i]);
        }
    }
    return 10000/(good_matches.size()+1);
}
```

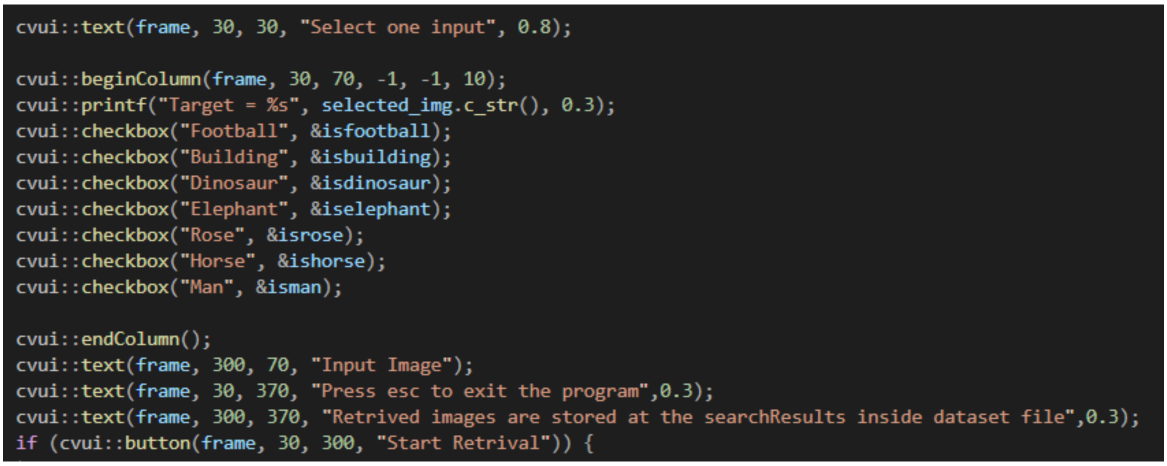
Being the most fastest algorithm among others, ORB is decided to be used to detect and compute the feature key points and descriptors. Fast Library for Approximate Nearest Neighbors (FLANN) is then used to find the good matches. The number of good matches found determines the similarity score obtained (the more the good matches found, the higher the similarity is).

CVUI

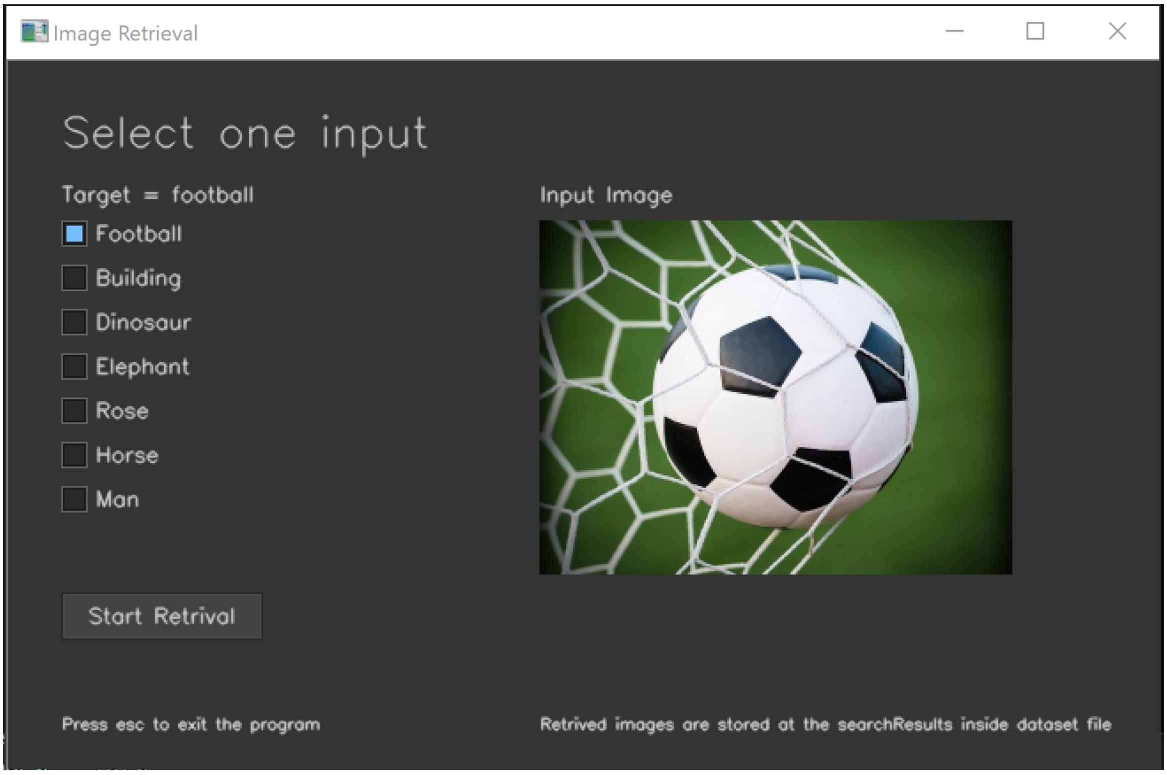


```
#define CVUI_IMPLEMENTATION
#include "c:\\cvui-2.7.0\\cvui.h"
```

CVUI 2.7.0 is used to implement the interface of the program



```
cvui::text(frame, 30, 30, "Select one input", 0.8);

cvui::beginColumn(frame, 30, 70, -1, -1, 10);
cvui::printf("Target = %s", selected_img.c_str(), 0.3);
cvui::checkbox("Football", &isfootball);
cvui::checkbox("Building", &isbuilding);
cvui::checkbox("Dinosaur", &isdinosaur);
cvui::checkbox("Elephant", &iselephant);
cvui::checkbox("Rose", &isrose);
cvui::checkbox("Horse", &ishorse);
cvui::checkbox("Man", &isman);

cvui::endColumn();
cvui::text(frame, 300, 70, "Input Image");
cvui::text(frame, 30, 370, "Press esc to exit the program",0.3);
cvui::text(frame, 300, 370, "Retrived images are stored at the searchResults inside dataset file",0.3);
if (cvui::button(frame, 30, 300, "Start Retrival")) {
```



Select one input

Target = football

☑ Football
□ Building
□ Dinosaur
□ Elephant
□ Rose
□ Horse
□ Man

Press esc to exit the program

Retrived images are stored at the searchResults inside dataset file

There are 2 main components the users can interact with: Checkbox, and button.

Checkboxes are implemented to let users set different target objects [football, building, dinosaur, elephant, rose, horse, man] to be retrieved from the database. The input image on the right is simultaneously updated as the template of the selected object.

Button is implemented to activate to retrieval program with the selected target object.

Output

Case - horse

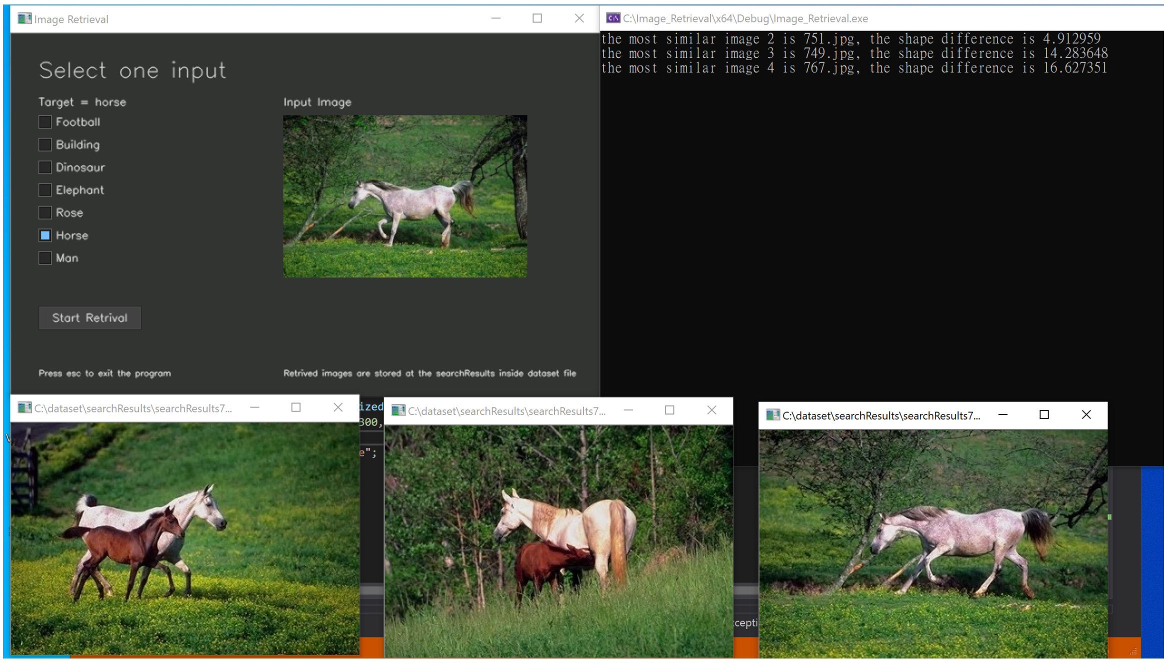


Image Retrieval
C:\Image_Retrieval\x64\Debug\Image_Retrieval.exe
C:\dataset\searchResults\searchResults7...
C:\dataset\searchResults\searchResults7...
C:\dataset\searchResults\searchResults7...

Select one input

Target = horse

- □ Football
- □ Building
- □ Dinosaur
- □ Elephant
- □ Rose
- ☑ Horse
- □ Man

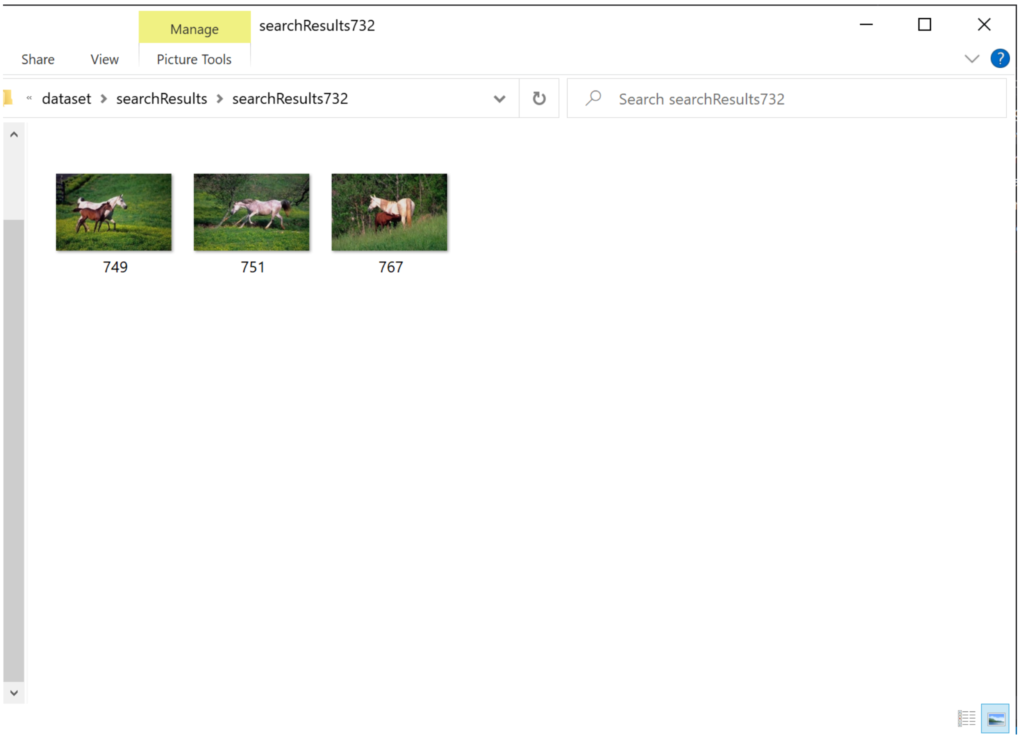
Input Image

Start Retrival

Press esc to exit the program

Retrived images are stored at the searchResults inside dataset file

```
the most similar image 2 is 751.jpg, the shape difference is 4.912959
the most similar image 3 is 749.jpg, the shape difference is 14.283648
the most similar image 4 is 767.jpg, the shape difference is 16.627351
```



The program does comparison between the selected template and images from database and generates an array of similarity score. The top 3 (without the template itself) scores will be displayed. The corresponded images will be retrieved to the searchResult file inside dataset file and be displayed to the screen.

Expected retrieved images

football

992

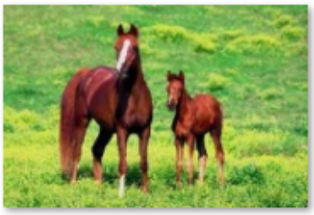
993

994

building

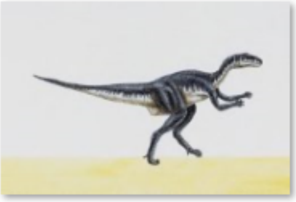
285

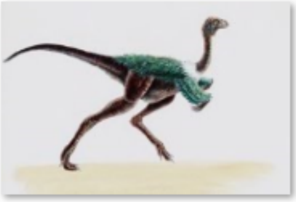
288

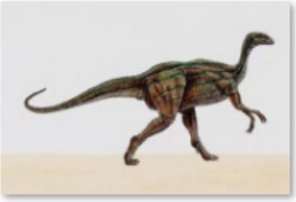


764

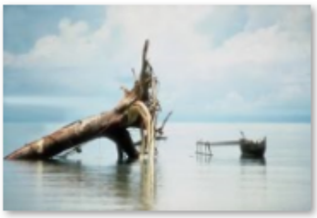
dinosaur

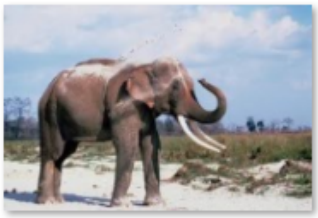

431

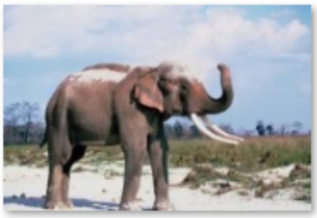

453


489

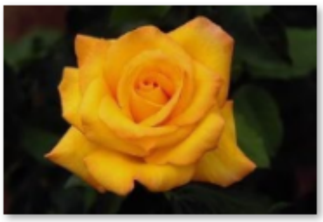
elephant

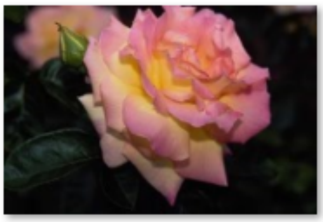

79

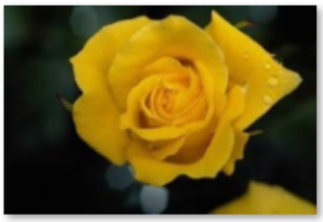

589


595

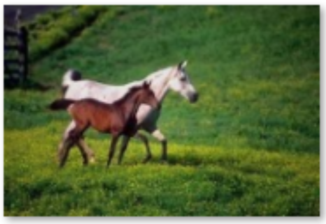
rose

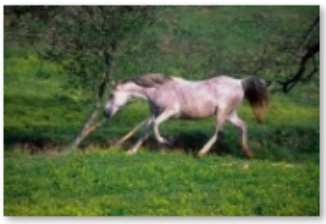

600

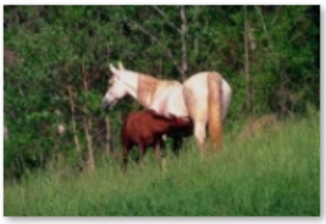

638


641

horse


749


751


767

man

Responsibilities of each group member

All the works in the project are contributed by Tam Chin Pang.