Fall | 10

The Impact of Using Different Video Compression Ratio for Face-feature Recognition

Si Chen

Dec 20, 2010

University at Buffalo-SUNY

Literature Review

The challenge of multimedia processing is to provide services that seamlessly integrate text, sound, and image and video information and to do it in a way that preserves the ease of use and interactivity. Now, the cameras are cheap and can be easily implement into TV, Computer, Notebook and cell phone. Therefore, how to use these cameras to enhance our life quality is a hot issue. Big IT companies create many applications to enhances human communications by exploiting both visual and aural senses and provide the ultimate flexibility in work and entertainment, like "Face time" on iPhone. However, the bandwidth rather than processing power remains the primary bottleneck for many complex multimedia application involving network communication. Current video coding algorithms use intelligent encoding to yield higher compression ratios. Compare with the lower compression ratio, the higher compression ratio may lead to bad image quality that may affect the accurate of object feature recognition. Therefore, I want to focus on measure the impact of using different video compression ratio for face-feature recognition. It is a good chance for me to get an in-depth understanding about digital video coding and analysis, feature extraction algorithms and software design.

I selected "Integration of Global and Local Information in Videos for Key Frame Extraction" as my major reference for the course project. In this paper, the author gives a brief introduction about how to extract the key-frame from one video and how to do feature vector extraction from video frames by using Scale Invariant Feature Transform (SIFT) method. This method transforms an image into a large collection of feature vectors. These feature vectors are invariant to location, scale and rotation, and robust to affine transformations and changes in illumination. Therefore, I can use this method to extract the face-feature from the original video. Here, I employ the idea in SIFT to further compare several videos with different compression ratio. Assuming that if the same object is detected in two different compression ratio videos, it means this object dose not affect by the compression process.

I've also learned how to evaluate the precision in this paper -- we need calculate the percentage of the extracted key frames and precision-hit deviation curves. In my opinion, the fewer key frames we can use to detected one object in two videos, the better.

So here is how my several steps looked like: obtain the original person facial video from camera. Compress this original video into different compression ratio and different parameters of video compression. Using Scale Invariant Feature Transform (SIFT) method to get the feature vector. Compare the compressed video with the original one, and evaluate the precision. Thus, I may find the optimal compression ratio, which can decrease the video size and have no effect to the face-feature recognition process (using SIFT method).

a. Introduction:

I. Advances in general multimedia communication research

The challenge of multimedia processing is to provide services that seamlessly integrate text, sound, and image and video information and to do it in a way that preserves the ease of use and interactivity. Now, the cameras are cheap and can be easily implement into TV, Computer, Notebook and cell phone. Therefore, how to use these cameras to enhance our life quality is a hot issue. Big IT companies create many applications to enhances human communications by exploiting both visual and aural senses and provide the ultimate flexibility in work and entertainment, like "Face time" on iPhone. However, the bandwidth rather than processing power remains the primary bottleneck for many complex multimedia application involving network transmission. Current video coding algorithms use intelligent encoding to yield higher compression ratios. Compare with the lower compression ratio, the higher compression ratio may lead to bad image quality that may affect the accurate of object feature recognition.
In the application of mobile augmented reality, where the video captured by user needs to match a set of images at the server, the image patches centered on key points represent crucial matching information. Similarities among these key point patches can enhance predictive encoding, resulting in a decrease in the transmission bitrate.
Encoding key point patches should have better matching results, due to the fact that the key points have already been located using the raw image. Trying to detect them on a degraded image should be less accurate.

II. Literature reviews

It has been shown that SIFT (Scale Invariant Feature Transform) descriptors can be used to achieve fairly robust object detection in still images [1]. SIFT has furthermore been used in several other related applications such as metric robot localization [2] and medical imaging [3]; and it was shown to be one of the best currently available descriptors in a comparative study [4].

In [5], the author discuss the effect of down sampling the face images on the accuracy reported for SIFT feature matching. They performed a number of experiments with different face image size. The result shows that the average number of SIFT features extracted decreases with decreasing the resolution of the image. [5] However, the author also gives a conclusion that the resolutions up to 50% give comparable result to those at full scale, while at 25% the accuracy decreases significantly.

III. Why you select this particular project

I want to focus on measure the impact of using different video compression ratio for face-feature recognition. It is a good chance for me to get an in-depth understanding about digital video compression coding and analysis, feature extraction algorithms and software design.
The conclusion of this project can be used in different applications. Like mugshot matching, surveillance, access control and personal identification, and law enforcement applications. [6] Since interaction is the future trend in TV, the result may guide the interactive system designer to build up a bandwidth efficiency application.

IV. Overview of the project

Project Goal

My project aims to analysis the impact of using different video compression ratio for face-feature recognition. I'm used one multimedia video file; the main part of this video is a human face. Then, I extract two frames from the video file and compressed the original video into different compress ratio and different file format. After that, I developed a program which can used the SIFT algorithm to match the face feature between the two extract frames and the compression video frames. And it can output the key point number for each frame and give a result to tell whether these two frames are similar. Therefore, we can compare the impact of different video compression ratio by analysis the matched frame number.

Algorithm

SIFT:

Scale Invariant Feature Transform (SIFT) features are features extracted from images to help in reliable matching between different views of the same object. [7] The extracted features are invariant to scale and orientation, and are highly distinctive of the image. They are extracted in four steps. The first step computes the locations of potential interest points in the image by detecting the maxima and minima of a set of Difference of Gaussian (DoG) filters applied at different scales all over the image. Then, these locations are refined by discarding points of low contrast. An orientation is then assigned to each key point based on local image features. Finally, a local feature descriptor is computed at each key point. This descriptor is based on the local image gradient, transformed according to the orientation of the key point to provide orientation invariance. Every feature is a vector of dimension 128 distinctively identifying the neighborhood around the key point. [5]

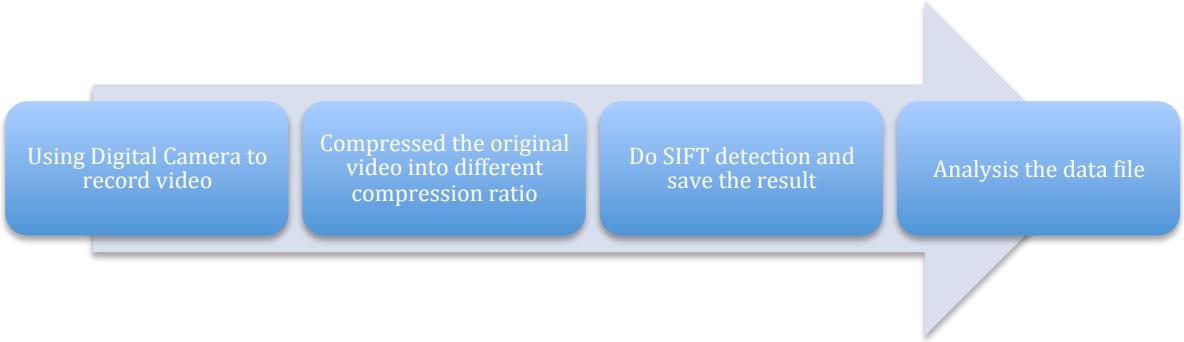
KD-forest:

An indexing technique, which allows efficient approximate k-nearest, neighbors search.

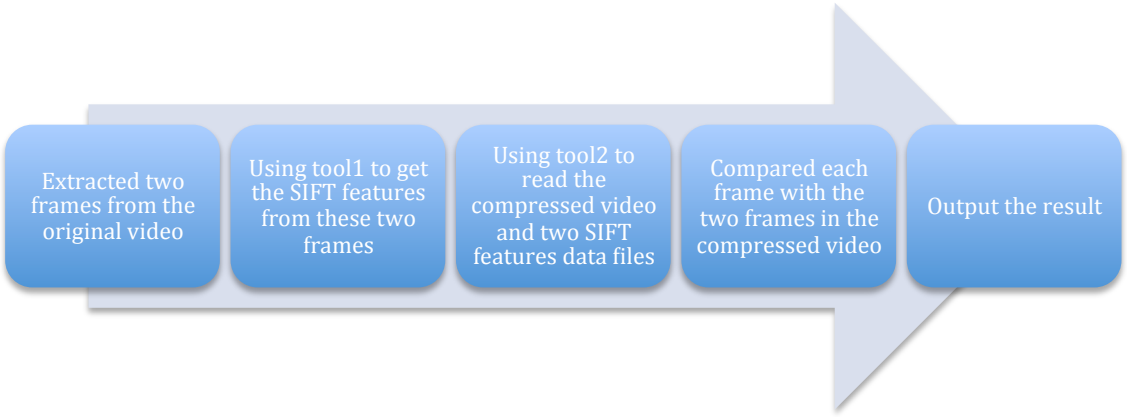
b. Your Proposed Approach:

I. The overall system and how the proposed component fits

Four different parts combine the overall system.
The first one is the video create system. I use my digital camera to record my face and create the original video and copy the video files into computer.
The second one is called the video file process system. This system can provide different compression ratio video file from the original multimedia video.



The third one is called the main process system, it can read the video file, processed SIFT detection and output the result.
The fourth one is the analysis system, which is a combination of different Linux tools. This system is can read the output data files from the third system and analysis it.
I propose an approach based on SIFT features for face recognition. The SIFT features are extracted from the original video. In this project, I dumped two SIFT features from the original video frames. One is the frame at the beginning of the original video that still remains in the compression video's frames. The other SIFT features is dumped from the last frame of the original video and it is not remains in the compression version because I cutoff 1/3 frames from original video when I compress it.



After dumped two SIFT features, I programmed another tools to match the SIFT features. This program will reads two SIFT features files and compared these features with the video frame by frame. For each frame, the program will run twice by using different octave number. If one frame is matching the original video frames

(using KD-forest algorithm to count the distance). It will output the message like "A shows!!!" or "B shows!!!"
The typical result is shown below:

```
~~~~~~~~~~start of octave~~~~~~~~~~~~
~~~~~~~~~~start of octave~~~~~~~~~~~~
Octave: 1
Percentage:0.653333
avg:66731.484375
threshold==0.250000
A shows!!!
Percentage:0.811688
avg:66608.929688
threshold==0.250000
B shows!!!
~~~~~~~~~~~end of octave~~~~~~~~~~~~
~~~~~~~~~~start of octave~~~~~~~~~~~~
Octave: 2
Percentage:0.166667
avg:72519.976562
threshold==0.050000
A shows!!!
Percentage:0.239448
avg:64910.890625
threshold==0.050000
B shows!!!
~~~~~~~~~~~end of octave~~~~~~~~~~~~
```

II. The hardware/software implementation plan

The experiment environment is Mac OS X 10.6.5 with processor 2.66GHz Intel Core i5 and 8 GB 1067 MHz DDR3 Memory.
In this project, I used C programming language with OpenCV and VLFeat library to read the media video files and processed SIFT detection. Xcode IDE was chosen as the primer coding and debugging experiment.
OpenCV is an open source computer vision library. The library is written in C and C++ and runs under Linux, Windows and Mac OS X.
The VLFeat open source library implements popular computer vision algorithms including SIFT, MSER, k-means, hierarchical k-means, agglomerative information bottleneck, and quick shift. It is written in C for efficiency and compatibility, with interfaces in MATLAB for ease of use. It supports Windows, Mac OS X, and Linux.
The debugging tool is gdb and the complier is gcc.
I also used sed, bash shell script and other Linux tools when processing data files.

III. Resource used for implementation

The input media file is capture by digital camera.
To provide different compression ratio movie file, I selected "Squeeze 6" software as the media process program. The Squeeze interface is designed to simplify the video compression workflow and provide easy access to the many power features available in Sorenson Squeeze. It support AVI, MPEG, MPG MOV and other 20

formats as the input formats, and it can output AVI, FLV, MOV, MP4 (H.264), MPG, SWF, VC1, WMV as the output formats.
I set different data rate to compress the video. Since SIFT algorithm is sensitive to the search region, to achieve a reasonable result, I set each compressed video file to the same resolution.
In the first experiment, I choose H.264 as the codec. I will try to detect the impact of different codec in experiment two.
To simplify the process steps, the original video contains 2-second blank screen, therefore, the key points number will be zero (or near zero) at that time. When analysis the video's key points number, we need delete several number related to the 2-second blank frame. And if the key point numbers become zero again, I assume that the video is finished and start over.

c. Outcome and Deviations

I. Presentation of the project outcome (experimental results)

	A	B	total	match A%	match B%	size(kb)	Total Bit rate	ratio
original	176	178	262	67.18	67.94	14336	21906	1.00
new	51	51	122	41.80	41.80	2355.2	8935	2.45
new1	51	52	122	41.80	42.62	1740.8	6752	3.24
new2	51	51	122	41.80	41.80	1433.6	5681	3.86
new3	50	50	122	40.98	40.98	553	2154	10.17
new4	50	52	122	40.98	42.62	410	1595	13.73
new5	52	52	122	42.62	42.62	291	1118	19.59
new6	52	52	122	42.62	42.62	229	890	24.61
new7	52	52	122	42.62	42.62	135	497	44.08
new8	51	48	122	41.80	39.34	74	276	79.37
new11	47	47	122	38.52	38.52	66	233	94.02
new12	32	32	122	26.23	26.23	57	209	104.81
new9	28	28	122	22.95	22.95	53	191	114.69

Table 1. Summary of the analysis of match percentage at different compression ratio

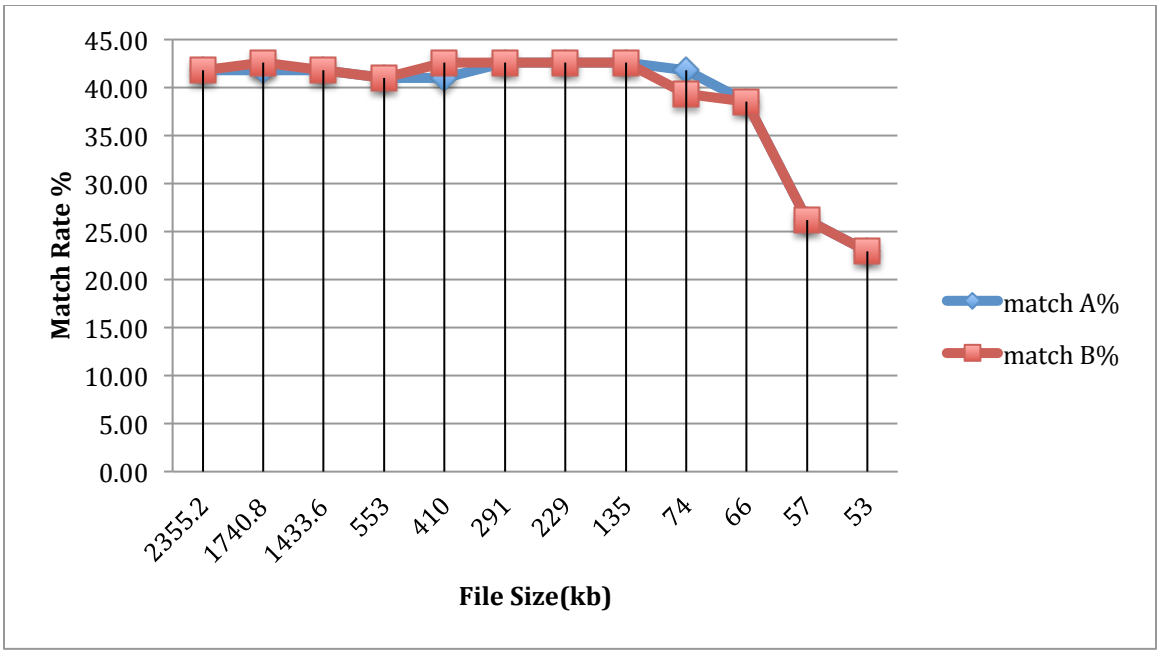


Fig.1. Match rate at different compression ratio

	A	B	total	match A%	match B%	size(kb)	
original	107	108	140	76.43	77.14	356	Original file H.264 640*480
new1	25	29	72	34.72	40.28	193	HD Quality 480p H.264 848*480
new2	39	36	72	54.17	50.00	213	DVD Quality H.264 720*576
new3	38	37	72	52.78	51.39	295	Optimal Quality H.263 352*288
new4	40	39	72	55.56	54.17	446	Highest Available Quality MPEG4
new5	38	36	72	52.78	50.00	160	Standard Quality H.263
new6	39	39	72	54.17	54.17	336	Balanced Quality MPEG4
new7	38	36	72	52.78	50.00	111	Smallest Size H.263
extra	59	59	82	71.95	71.95	217	H.264 320*240

Table 2. Summary of the analysis of match percentage at different compression methods

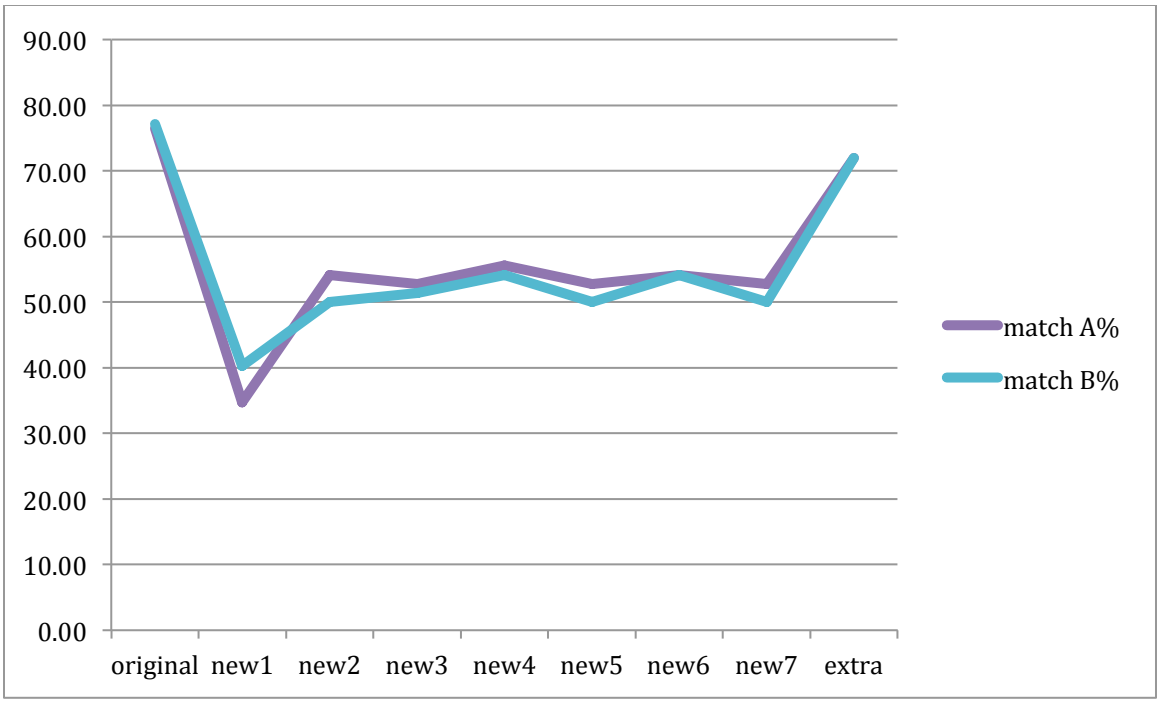


Fig.2. Match rate at different compression methods

II. Discussion of the outcome (any deviation? Why?)

Fig 1 shows that the percent of matched frame in different compression ratio remains stable until the file size drop below 135Kb.

This results show that the SIFT algorithm is very robust. Even if the compressed file is only 1/50 times as the original file, we can still get comparable result to the original one.

However, if the compression ratios surpass 100, the accuracy decreases significantly. The reason is that the video frames is compressed too much and become blur, the key features begin to loss. Therefore, the SIFT algorithm cannot detect the frames correctly.

Fig 2 shows that different compression method may lead to different match accuracy.

III. Lessons learned from the experiments

From the result, we can assume that in each compression method, there is a SIFT detection threshold. If we not reach the threshold, the match result will remains stable and acceptable. However, if the compression ratios surpass the threshold, the accuracy decreases significantly.

If we want to design an on-line face recognition system and the bandwidth of is limited. We can choose SIFT algorithm to do face recognition and we must measure the SIFT detection threshold. The data compression ratio should not surpass the threshold.

d. Summary and Discussion

I. Summary of the project

This project shows that there remains a threshold when we using SIFT algorithm to do Face-feature Recognition with video file. In this project, I test two different human face videos and compressed them into different bit rate. The result is reasonable and the output data is accurate.

However, the experiment sample size is still too small and the result only suitable for the MOV format files. We need more experiment to promotion of this conclusion and given more accurate quantitative interpretation.

II. Lessons learned in this course

In this course, I've learned a lot of data compression methods, many design issues related to achieve to high QoS, effective dissemination multimedia over heterogeneous IP wireless broadband networks and a little digital rights management and adaptation schemes. The most impress part of this course is the introduction of Peer-to-peer (P2P) Systems, the different prototypes (tree-push, mesh-pull and push-pull) of P2P streaming systems give me much inspirations in my own wireless sensor networks (WSN) design (which I decided to do related research in these area to graduate). As an EE student, the introduction of different wireless technology is not hard to understand and it is very useful. The homework set is not easy; to finish these homework sets I need to read the textbook over and over again and "google" several conceptions. Since I just self-learned little algorithm analysis before, the homework set 3, problem 2 seems very hard to me and I have to teach myself how to measure and calculate the complexity of each algorithm.

As for the project, I've reviewed several concept related compression ratio. One of the confusing parts is how to prepare different compression ratio video file. To give a reasonable result, I redesign the experimental process again and again.

e. Acknowledgement

Thanks to professor Changwen Chen and TAs.

f. Reference List

[1] Y. Ke and R. Sukthankar, “PCA-SIFT: A More Distinctive Representation for Local Image Descriptors,” *Proc. CVPR*, vol. 2, pp. 506–513, 2004.

[2] D. Lowe, “Object recognition from local scale-invariant features,” *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, vol. 2, pp. 1150–1157, 1999.

[3] S. Se, D. Lowe, and J. Little, “Vision-based mobile robot localization and mapping using scale-invariant features,” *Robotics and Automation,*

2001. Proceedings 2001 ICRA. IEEE International Conference on,
vol. 2, 2001.
[4] M. Moradi, P. Abolmaesoumi, and P. Mousavi, "Deformable Registration Using Scale Space Keypoints," *Proceedings of SPIE*, vol. 6144, p. 61442G, 2006
[5] Mohamed Aly. Face Recognition using SIFT Features. *CNS186 Term Project* Winter 2006
[6] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld. Face recognition: A

literature survey. *ACM Computing Surveys, 35(4):*399–458, December 2003.
[7] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60, 2004.