**COURSEWORK ASSIGNMENT** **UNIVERSITY OF EAST ANGLIA**
**School of Computing Sciences**

**MODULE: CMP-6035B - Computer Vision**
**ASSIGNMENT TITLE: Image categorisation II.**

| | | | |
|---|---|---|---|
| **DATE SET:** | **Mon of week 9** | | |
| **DATE & TIME OF SUBMISSION:** | **15:00 Wednesday of week 14** | | |
| **ASSIGNMENT VALUE:** | **40%** | | |
| **SET BY:** | **MM** | **SIGNED:** | |
| **CHECKED BY:** | **GDF** | **SIGNED:** | |

**Aim:**
To reinforce material presented in the lectures and to give you practical experience programming computer vision algorithms in Matlab.

**Learning outcomes:**
An in-depth understanding of the related computer vision algorithms. An understanding of the algorithm evaluation process. Improved Matlab programming skills with a focus on creating efficient solutions.

**Assessment criteria:**
Marks will be awarded for the electronically submitted code and the report. Further details are provided after the Problem Statement.

**Description of assignment:**
See the attached Problem Statement.

**Handing in procedure:**
This is a group exercise that should be done in pairs. You should continue to work in the same pair you worked during coursework 1, unless there is a problem, in which case you should contact me.
The source code should be zipped into a file with the name containing Student IDs of both members of the group. **Both** members of the group must then submit this (same) file to the digital dropbox located on Blackboard page in the Coursework 2 folder. **Make sure you test your solutions on a lab machine BEFORE you submit**.
The report should be submitted electronically through the E-vision pages of the student profile. Again **both** members of the group must submit (the same) report.

Late submissions need to follow the appropriate late hand-in procedure. You can find out this information from the Science Teaching Office.

If you have medical or other problems you can seek extensions to coursework deadlines. However, it is essential you obtain proper documentation in such cases (i.e. a medical certificate), to be handed in to the School General Office at the time of the difficulty.

**Plagiarism:**
Plagiarism is the copying or close paraphrasing of published or unpublished work, including the work of another student without the use of quotation marks and due acknowledgement. Plagiarism is regarded a serious offence by the University and all cases will be reported to the Board of Examiners. Work that contains even small fragments of plagiarised material will be penalised.

# Problem Statement

In coursework 1 you carried out a literature review of the image classification research and implemented a benchmark image classification algorithms based on tiny image and colour histogram features; and a kNN classifier. In this coursework you will use the knowledge you have gained in coursework 1 and from the lectures to implement further improvements to your algorithm.

You will work with the same image dataset. You will use the same functions as a skeleton for your implementation.

There are many ways one can approach this coursework task. In coursework 1 you have found that tiny images and naive colour indexing do not support accurate classification. During the lectures, you have learned about other image features, classifiers and meta-algorithms that could be applied in this problem e.g. SIFT, HOGs, bag of features, spatial pyramid kernels etc. You have also performed your own literature review into these and similar algorithms. You will need to implement some of these to build a better performing solution. You may discover that your literature review in coursework 1 was insufficient and you need to consider other algorithms. You are encouraged to read more.

One of the algorithms you will need to implement during this coursework is bags of quantised SIFT features. First, you will need to establish a dictionary of visual words. This can be achieved by collecting large number of SIFT features from the training set and then clustering them with *kmeans* algorithm. The number of kmeans clusters is the size of your dictionary and also the size of your features. You might start by clustering many SIFT descriptors into e.g. $k = 50$ clusters. The 50 features will partition a 128 dimensional SIFT feature space into 50 regions. Any new SIFT feature in the test set can be associated with one of these regions by finding the closest region centroid.

Since extracting and clustering SIFT features can be slow, we suggest that you precompute the cluster centroids and store them in a .mat file, which will save you lots of time in subsequent runs of the algorithm when you want to test different settings of the classifier. See `build_vocabulary.m` for more details. You can find the updated skeleton implementation for this coursework in the `coursework2_starter.m`

Once you have the cluster centroids, you can represent images as histograms of visual words. From each image you will extract 100s of SIFT features (use dense sampling provided in `vl_dsift`) and count how many of them fall into each bin in your visual word dictionary. This can be achieved by searching for the nearest dictionary centroid for every SIFT feature in an image. Hence, if you have a dictionary of 50 visual words, and detect 200 SIFT features in an image, your bag of visual words will be a 50 dimensional histogram where each bin counts how many times a SIFT feature was assigned to that cluster and sums to 200. You should normalise this histogram so that it sums to one to ensure that image size does not affect the magnitude of the bag of SIFT feature. See `get_bags_of_sifts.m` for more details. There are a few parameters for the bag of SIFT representation (number of clusters and other SIFT parameters). You will need to tune these parameters. See the documentation for the `vl_dsift` function.

The next task is to train 1-vs-all linear SVM to operate in the bag of SIFT feature space. This classifier has been explained in the lecture notes. Since linear classifiers such as this one are binary, you will need to train 15 binary, 1-vs-all classifiers. 1-vs-all means that each classifier will be trained to recognize 'forest' vs 'everything-but-forest', 'kitchen' vs 'everything-but-kitchen', etc. You will have to evaluate all 15 classifiers on each test feature and the classifier which is most confident about the positive result will determine its class label. SVM has a regularisation parameter called $\lambda$ or $C$. You have to tune this parameter over a wide range of values as your results may strongly depend on it. See `svm_classify.m` for more details.

To complete the assignment you should:

1. Implement bag of words feature extraction utilising SIFT features (grayscale). You should use kNN classifier from coursework 1 and you need to implement an additional classifier - linear SVM. Test the algorithm for various parameters and discuss the results.

    - The same as above, but for the colour images. You will need to devise a way of fusing colour information and SIFT features. Look at [1] for some ideas.

2. Repeat 1. for the spatial pyramids with SIFT features, see [2].

3. Implement any other scene recognition algorithm. Here, you can use any algorithm from the literature as well as any classifier. You can also devise your own algorithm or fuse some of your ideas with the existing algorithms.

**Additional information**

1. Report and code hand-in electronically, **Wed 3pm Week 14**. Report through E-vision, code through Blackboard (link in the Coursework 2 folder). Both members of the group must submit both files.

2. At the end of the report you must state the proportion of the contributions of the group members. This needs to be only one sentence.

3. The functions implementing your algorithm has to be called according to the specified naming scheme and placed in a folder named with your registration number. Any additional functions must be placed in the same folder.

4. After copying your coursework folder into my machine, I should be able to run your code on my machine as is. The only edit to the code you may expect me to make is changing the `data_path` variable in `coursework2_starter.m`.

**Installation of the dataset and the starter code.**

You need to install a VLFeat MATLAB toolbox [1]. Download the toolbox and run `vl_setup`. The toolbox contains a large selection of computer vision related functions. You must not bundle this toolbox with your submission. You can assume that on my machine the toolbox file path is set in MATLAB.

# Report

Your written report should be no longer than 11 pages in length. Your report should **not** include any Matlab code (algorithms can be described as mathematical equations and/or pseudo code). All pages should have reasonable margins and a font size of 11pt should be used. A part of a report that is over the 11 allowed pages will not be marked. Make sure you include any relevant citations in your report. For reference you are pointed to the statement on plagiarism in the unit description. Make sure that your report contains informative clearly labelled figures describing your results.

# Marking Scheme

This coursework carries 40% of the module weight, therefore you may receive up to 40 marks for this coursework. The marks will be awarded according to the following scheme:

1. Implementation, testing and reporting of the Bag of SIFT features:
   (a) kNN and grayscale SIFT. (**5 marks**).
   (b) SVM and grayscale SIFT. (**5 marks**).
   (c) kNN and SIFT+colour. (**4 marks**).
   (d) SVM and SIFT+colour. (**4 marks**).

2. Implementation, testing and reporting of the spatial pyramids method with SIFT features:
   (a) kNN and grayscale SIFT. (**3 marks**).
   (b) SVM and grayscale SIFT. (**3 marks**).
   (c) kNN and SIFT+colour. (**3 marks**).
   (d) SVM and SIFT+colour. (**3 marks**).

3. Implementation, testing and reporting of the additional algorithm of your choice. **10 marks**
   - Here, marks will be awarded according to the complexity, performance and the thoroughness of implementation, testing and reporting.

[1] `http://www.vlfeat.org/install-matlab.html`

# References


[1] K. van de Sande, T. Gevers, and C. Snoek, “Evaluation of color descriptors for object and scene recognition.” in *Proceedings of CVPR*, 2008.

[2] S. Lazebnik, C. Schmid, and J. Ponce, “Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories,” in *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2*, ser. CVPR ’06. Washington, DC, USA: IEEE Computer Society, 2006, pp. 2169–2178. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2006.68