Computer Science Part II Project Proposal

Video Steganography using Motion Vectors

E. Liberis, Homerton College

el398@cam.ac.uk

20th October 2015

Project originator: E. Liberis

Supervisor: Daniel Thomas
Director of studies: Dr. Bogdan Roman
Overseers: Dr. D J Greaves, Prof. J Daugman

Introduction and Description of the Work

Steganography refers to a set of techniques for concealing information within seemingly innocent carrier data (covers), and for detecting such hidden information. This is widely used for enabling communication where the detection of the existence of the message would attract unwanted attention [1].

Files with commonly-found data formats are less likely to raise suspicion as covers, and redundancy in the structure of the data provides opportunities to hide data. Digital media tends to have both of these properties, contributing to their widespread use as covers in earlier work [2, 5, 3, 4]. This project will be investigating the use of motion vectors in video steganography.

Successive frames in a video often only differ by the movement of pixels in some regions of the frame. This "temporal correlation" is exploited by modern video codecs, notably H.264, to improve compression ratios by storing only the motion data since the last frame, where doing so is cheaper than storing the whole frame.[1] Frames stored in this way are called P- or B- frames and are represented by a list of blocks of pixels that have moved since the previous frame (*macroblocks*), and the direction of the movement (*motion vector*) [6]. Computing motion vectors is a tradeoff between accuracy and computation time, so codecs typically settle for an approximation instead of an optimal MV. While MVs are encoded losslessly, this accuracy tradeoff allows minor changes to them go unnoticed [3]: a property essential to the steganographic techniques underlying this project.

The main deliverable of this project will be a steganography tool for embedding arbitrary data into H.264-encoded videos, using a variety of motion-vector-related techniques. A common trick for reducing distortion of the cover data is to modify the least-significant bits of encoded values, be they LSBs of pixels in an image or values of a motion vector. As well as naïvely embedding the message in LSBs, there are some more intelligent approaches for making messages harder to detect, such as selecting target LSBs using a PRNG or avoiding regions with low variance [2]. Methods that specifically target motion vectors have been proposed by Zhang *et al.* [5] and Fang *et al.* [3], which make use of length, phase and inaccuracies of a motion vector.

The branch of steganography concerning the detection of embedded messages is called steganalysis. Another deliverable of this project is to provide a toolkit to perform video steganalysis that neatly integrates with existing scientific computing packages (Matlab, Python-based libraries). A popular approach in steganalysis is to look at statistical differences of modified and unmodified values, as selectively modifying certain values may make them stand out from the rest. General approaches use histograms, Chi-Squared attacks [2], and motion-vector-specific attacks (Deng et al. [4]), exploit correlations seen in neighbourhoods of MVs and other statistical metrics.

I propose a project, consisting of two main deliverables:

[1] Codec periodically inserts individually encoded frames (I-frames) whenever picture changes significantly and at regular intervals (to recover if a streaming error occurs).

- Development of an application that allows end user to encrypt[2] and embed their secret message into H.264 video files, using embedding algorithms introduced above.
- Development of a steganalysis suite that implements some of the recent research advances and integrates into scientific computation packages.

Resources Required

I will be using C++ as a main language for developing the steganographic tool, primarily due to my experience with it and the speed and ease of interfacing with codecs: necessary for manipulating motion vectors. Steganalysis tools will be developed using Matlab or Python, as both have extensive support for scientific computation and are popular within the community.

I will be using my own computer, running Ubuntu 15.04, for convenience. The project can be continued on MCS machines should it become unusable. I accept full responsibility for this machine and I have made contingency plans to protect myself against hardware and/or software failure. Nightly backups will be made to a removable storage and `Dropbox`, and the code will be hosted on `GitHub` (with `git` version control).

No other special resources are required.

Starting Point

This project uses some of the cryptography concepts introduced in Part IB *Security I* course. I have familiarised myself with the general concepts of steganography and LSB embedding though some introductory texts and relevant papers, and briefly looked at H.264 codec format.

Implementing my own H.264 codec is out of scope for this project, so a possibly modified version of a library (provisionally `libx264`) will be used to gain access to the motion vectors. Steganography tools may also rely on some inbuilt routines of scientific computing packages (plotting, histogram, matrices handling, etc.).

The project may also benefit from material in Part II courses *Information Theory*, *Digital Signal Processing, LaTeX and Matlab* and *Security II*.

Substance and Structure of the Project

The aim of this project is to (a) develop a steganography tool, and (b) develop a steganalysis suite, as described in the first section.

Both aims will require further research into H.264 codec and ability to modify MVs of a H.264-encoded video file, which would in turn require parsing the format of the codec, modifying motion vector values, and repackaging the data back into a playable video file. As developing a H.264 codec does not relate to steganography and is potentially error prone, I

[2] Since properly encrypted data is indistinguishable from random noise encrypting the secret message thwarts many statistical attacks which would otherwise detect patterns in the data.

plan to leverage existing libraries and/or tools, such as `ffmpeg`, to achieve this. As codecs are typically written in C++, it is a natural choice for this task.

Modifying MV requires further research into steganographic techniques, as well as the implementation and adaptation of these algorithms for use in a codec. The application should also be able to take user data and perform encryption prior to embedding, which would require using a relevant cryptography library.

The second objective will require substantial further research of steganalysis methods described in recent academic papers, as well as an investigation of how to use and integrate scientific packages for implementing these methods.

The dissertation shall consider the embedding techniques, evaluating their relative applicability, detectability, embedding capacity, speed, effectiveness, and resistance to attacks by developed steganalysis methods. Detectability will also be evaluated by conducting a study with human participants, asking them to distinguish between a modified and an unmodified video. Effectiveness of steganalysis methods will be evaluated by comparing their success in attacking implemented methods on a dataset of videos.[3]

The steganography application will be manually and automatically tested to ensure it works as reliably as expected. Individual modules will be tested using unit tests and application as a whole using integration tests. Development will follow the iterative (spiral) model, continuously adding embedding techniques and steganalysis methods, and `gitflow` workflow.

Success Criteria

The project shall be considered successful if steganography application, steganalysis tools and/or accompanying disseration (as appropriate) satisfy the following requirements (prioritised using the *MoSCoW* rule):

- *(M)* Ability to modify motion vectors of a H.264 video file.
- *(M)* Multiple LSB and non-LSB MV embedding techniques.
- *(M)* Multiple steganalysis methods.
- *(S)* Encryption of the secret message prior to embedding.
- *(M)* Explanation and justification of changes to or inapplicability of some techniques, if required.
- *(S)* Compatibility with existing scientific computation packages (*Matlab* or Python-based packages)
- *(M)* Evaluation of the following properties of techniques:
  - *(M)* Applicability to MV steganography.
  - *(M)* Resistance to attacks.
  - *(C)* Embedding capacity.
  - *(C)* Speed of processing videos.

[3] Yet to be obtained, but that should not pose any problems.

  - *(S)* Detectability by a human.
- *(M)* Evaluation of the following properties of methods:
  - *(M)* Effectiveness against implemented embedding techniques.
  - *(S)* Usefulness (verbosity, amount of information provided) to the steganalyst.
- *(M)* Introduction of the underlying theoretical background.

Extensions

- Deng et al. [4] propose using statistical differences between videos with and without embedded messages as features for a classifier. Using this and other features proposed in similar works, a classifier trained using machine learning could be produced.
- Currently, social networks such as YouTube, Facebook and Tumblr transcode (convert between formats) user-uploaded videos before presenting them to other users. This results in MVs being recomputed and overwritten, destroying the hidden message. More aggressive embedding techniques could be developed, possibly using some control data and error-correction codes, to withstand such transcoding.

Timetable and Milestones

Weeks 1 – 2 (Oct 22 – Nov 5)

- Research and implementation of access and modification of MVs in a H.264 video file.

Weeks 3 – 4 (Nov 5 – Nov 19)

- Background reading on steganography and other relevant preparation.

Weeks 5 – 6 (Nov 19 – Dec 3)

- Investigation and implementation of popular image steganography techniques, including LSB embedding, and evaluation of their relevance to MV steganography.

Weeks 7 – 8 (Dec 3 – Dec 17)

- Creation of tools that allow to carry out statistical attacks on aforementioned methods and their evaluation.
- Research and implementation of the methods of video watermarking proposed by Zhang *et al.* [5] and Fang *et al.* [3]

Weeks 9 – 10 (Dec 17 – Dec 31)

- Finishing implementation of aforementioned methods.
- Christmas vacation and catch-up (if required).

Weeks 11 – 12 (Dec 31 – Jan 14)

- Research and implementation of statistical steganalysis methods.
- Evaluation of created tools against all implemented embedding algorithms.

Weeks 13 – 14 (Jan 14 – Jan 28)

- Finishing any remaining implementation.
- Writing progress report and preparing for the presentation.

Weeks 15 – 16 (Jan 28 – Feb 11)

- Evaluation of the indistinguishability of videos with hidden messages and ordinary videos using human test subjects.
- Remaining evaluation tasks.

Weeks 17 – 18 (Feb 11 – Feb 25)

- Any remaining catch-up work.
- Implement extension tasks, if time permits.

Weeks 19 – 20 (Feb 25 – Mar 10)

- Commence writing dissertation.
- Ongoing evaluation tasks and further exploration of extension goals.

Weeks 21 – 22 (Mar 10 – Mar 24)

- Write “Introduction” and “Preparation” chapters.
- Request supervisors’ feedback and iterate upon it.

Weeks 23 – 24 (Mar 24 – Apr 7)

- Write “Implementation” chapter.
- Request supervisors’ feedback and iterate upon it.

Weeks 25 – 26 (Apr 7 – Apr 21)

- Write “Evaluation” chapter.
- Request supervisors’ feedback and iterate upon it.

Weeks 27 – 28 (Apr 21 – May 5)

- Finish remaining parts of the dissertation.
- Polish and incorporate final feedback.
- Submit the dissertation.

References

[1] Omar J. Pahati *"Confounding Carnivore: How to Protect Your Online Privacy,"* AlterNet, Nov 28, 2001 (retrieved Oct 2015)
http://www.alternet.org/story/11986/confounding_carnivore%3A_how_to_protect_your_online_privacy

[2] Philip Bateman *"Image Steganography and Steganalysis,"* 2008
http://chemistry47.com/PDFs/Cryptography/Steganography/Image%20Steganography%20and%20Steganalysis.pdf

[3] Ding-Yu Fang; Long-Wen Chang, *"Data hiding for digital video with phase of motion vector,"* IEEE International Symposium on Circuits and Systems, 2006. Proceedings.
http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1692862&isnumber=35661

[4] Yu Deng, Yunjie Wu, Linna Zhou, *"Digital video steganalysis using motion vector recovery-based features,"* Appl. Opt. 51, 4667-4677 (2012)
http://www.ncbi.nlm.nih.gov/pubmed/22781241

[5] Jun Zhang; Jiegu Li; Ling Zhang, *"Video watermark technique in motion vector,"* in Computer Graphics and Image Processing, Proceedings of XIV Brazilian Symposium on , vol., no., pp.179-182, Oct 2001
http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=963053&isnumber=20786

[6] International Telecommunication Union (ITU), *"H.264: Advanced video coding for generic audiovisual services,"* Feb 2014 (specification of the H.264 video codec)
https://www.itu.int/rec/T-REC-H.264-201402-I/en

Study involving Human Participants

Description of the study

The aim of this study is to determine whether using steganographic methods to hide data in video files results in visible changes to the video. The experiment will be conducted as follows.

In each trial, participants will be shown pairs of ostensibly identical videos, positioned side-by-side, one of which has been modified to contain a hidden payload. They will choose the video which they think was modified, after which the true answer will be revealed.

The experiment will feature 7-10 pairs of videos, each up to 30 seconds long, taking up to 20 minutes per participant. Which video is the modified one will be randomised, to avoid positional bias. Since the steganographic methods rely on motion data encoded in the videos, videos containing a wide range of movement, from almost still to highly dynamic, will be used.

We hypothesise that human participants will not be able to distinguish between modified and an unmodified video. To test this, we will determine if the data support the opposite claim to a statistically significant level – that humans *are* able to tell the videos apart. We will look into the numbers of times the participants were right or wrong in their choice and test whether there is a statistically significant skew in either direction.

Precautions to be taken to avoid any risk

No personal data about the participants will be collected and all results will be recorded anonymously. Videos shown to the participants will be free from flashing images, emotionally neutral, and free from distressful, disturbing, offensive or otherwise unsettling images.

Resources Declaration

The project does not require any special resources except own laptop, for convenience.

Specifications:

- Intel(R) Core(TM) i5-3210M CPU @ 2.50GHz
- nVidia Geforce GT 630M 2GB
- 500GB HDD
- 4GB RAM

I accept full responsibility for this machine and I have made contingency plans to protect myself against hardware and/or software failure.

The project can be continued on MCS machines should the laptop become unusable. Nightly backups of the project files will be made to a removable storage and `Dropbox`, and the code will be hosted on `GitHub` (with `git` version control).