Project

Spring 2017
(Last update: January 6, 2016)

Background

Project requirements: This course requires a project to be performed by one or two students, requiring about 40 hours effort *per person* over the entire semester. A project includes the following phases: choosing a project topic, writing a project plan, conducting the project, writing a mid-term and final project report, and presenting the project in class. As a team, you may choose to submit one report or two. If you submit one report, indicate clearly which sections were written by each team member.

Project purpose: The project provides hands-on experience implementing algorithms for video processing, defining experiments, and exploring algorithmic performance in the context of specific applications.

Project deliverables

February 16: Plan

March 23: Midterm report

April 28-30: Oral presentation

April 30: Final report

Project content: A project requires reading papers in a chosen area, implementing at least one algorithm in software, comparing performances of different approaches, and ideally proposing your own algorithms/improvements. The project topic should be distinct from any research you are doing or have done before. Your software can implement someone else's algorithm or your own. However, you must write something novel; it is **NOT** sufficient to merely download software from the web. You should acquire video that is suited to your specific project and test your system on your personal video as well as any relevant video from others. Your performance metrics should be carefully considered and appropriate for your study.

Project learning objectives: Learning objectives are divided into 3 categories: creating, evaluating, and analyzing. Be sure your project covers at least one aspect of each category.

- Create
  - Design an experiment (to compare one algorithm against another)

    - Create (an application scenario to test the method)

- Evaluate
    - Determine (performance of system on real videos)
    - Critique (the weakness of an algorithm)
    - Identify (assumptions made by the algorithm)
    - Select (one of several options and justify)

- Analyze
    - Classify (different potential algorithms based on their approach)
    - Predict (failure cases of an algorithm)
    - Explain (why a method was unsuccessful)

Deliverables Details

Project plan: The project plan is due **2/16/2017**. The project plan should include the following:

- Project title and team members
- An abstract: in one paragraph or two, describe briefly the project scope and what you hope to accomplish. See the excellent one-page description of how to write a technical abstract at

  `https://cbs.umn.edu/sites/cbs.umn.edu/files/public/downloads/`
  `Annotated_Nature_abstract.pdf`

- Project schedule: It should include a list of subtasks that you plan to accomplish (i.e. your deliverables); describe which member of your team is primarily responsible for each subtask and the deadline for completing it. You should allocate time for searching the project subject, initial literature search, preparing the midterm report, final report, and final presentation, in addition to the reading, simulation and software development you plan to do.
- List of references: this includes papers and documents and possibly web links that you have found that are relevant to your project topic. Please include complete citation of each item: author, title of the article, journal or book name, and if appropriate the URL and the date downloaded. However, do not cite Wikipedia. You should follow the standard citation format of IEEE journals.

You can discuss with me about the possible topics, either after class, arranged office hours or through email.

Midterm project report: The midterm report is due **3/23/2017**. The mid-term project report is an on-going document that starts from your project plan and will evolve into the final report. Include the following in your midterm report, in a way that reflects your progress to date.

- Project title and team members
- An abstract

- Project status. This can be divided into multiple sections. Section 1 should be an introduction or overview, which includes a **literature review**. Then each subtask can be a subsection. For each subtask, describe what you have accomplished so far, and what remains to be done. Include a status report of your software development and what videos or classes of videos you will run your software on.
- Revised Project Schedule: you may want to revise the original plan based on your progress so far. You may need to modify the deadlines only, or you may even need to modify the sub-tasks. The latter may be true if after your study so far, you want to redirect your project in terms of the technical content.
- List of references: you may add new references and delete irrelevant ones that originally appeared in the project plan.

Final report: The final report is due **4/27/2017**. The final report is an extension of your midterm report and should include the following, modified to reflect all you have accomplished. It should be between 5-10 page.

- Project title and team members
- An abstract
- Overview of the project and schedule (describe the subtasks and actual completion schedule. Specify who was responsible for which subtask if you had multiple members on the team.)
- Project accomplishment: This can be divided into multiple sections, each section describing one subtask that you have worked on. For each subtask, describing what you have done, what you have learnt. If your project includes the comparison of different systems/techniques, you may have separate sections describing each system/technique and another section comparing these systems/techniques and summarizes the pros and cons of each. You may have another section describing your proposed solution on how to modify each or combine them.
- Summary: This section should briefly summarize your findings: what worked and what didn't. It should also include a discussion of future work that you think should be studied.
- List of references: update from your midterm report as appropriate.

In your report, be sure to describe the whys of your choices. Treat your experiments as hypotheses to be tested, and describe your results as support for or against your hypothesis. Describe your basic real-world problem, and describe how you formalized it into a mathematical framework that could be implemented. Clearly state your assumptions. Describe how the failure-cases of your method might indicate the validity of your assumptions. Describe any optimization strategy you applied.

On writing and attribution

Comments on writing: English may not be your native language. This does not affect your ability to organize your thoughts and communicate them in a coherent fashion. Make it clear what you did and why you did it. An accurate and clear description is more important than perfect

grammar (although that?s important too!). **Be sure that the words that you write are you own.** If they are not your own words or your own ideas or your own software, you need to give the person who did write it or create it proper credit, through references and citations. If you use software you developed for another project (for another professor, or for another class), you must also acknowledge this.
If you plagiarize, you may receive a zero for the project or a failing grade for the class. All team members are responsible for any plagiarism in the final submission. You also should not plagiarize because it creates an unfair playing field for your fellow students, and it robs you of your chance to learn.

Additional tips

- Choose something that is both interesting and doable in the 40 hours. Consult with me for help refining your topic.
- Set intermediate goals. Be sure to have a fall-back plan if you are unable to meet your envisioned goals.
- One strategy is to choose a task from the literature that seems within reach and implement it. Your innovation can be small adjustments to the algorithm, and thoughtful experimentation. Another strategy is to choose an application that hasn't been done, and implement tried-and-true methods for another application to your new application.
- Formalize the problem by taking the initial idea and turning it into something that can be computed, evaluated, and explored.
- If you need data, there is often freely available datasets online. Data collection is a huge task, so best if your project does not need it.
- Design experiments that illustrate the your method, both the application you want as well as the performance of your algorithm. Quantity of results is not as important as thoughtfulness of the experiments and what those experiments can demonstrate.
- Run your on your own data to see how robust the method is to different types of videos. Try to create videos that break your algorithm.

Possible topics (last updated 1/9/17)

- 3D and stereo processing (stixels)
- Analyzing first person videos
- High Dynamic Range video
- Super-resolution from video
- Saliency estimation
  - Saliency based video coding
  - Saliency based video quality monitoring
- Eye motion and eye tracking
- Object tracking (beyond what's in class)
- Activity recognition (beyond what's in class)
- Video temporal segmentation
- Sports applications
- Video analytics using compressed videos
- Video copy (piracy) detection
- Video stabilization (beyond what's in class)
- Privacy protection in video
- Video forensics
- Scene segmentation
- Video transmission and error resilience
- Joint source channel coding
- Multi-camera video systems
- Egocentric video analysis