Final Project

# Introduction

For this project, you may work alone or with a partner to select a problem relevant to computer vision, implement the solution, and discuss the results of your efforts with both an in-class presentation and a written report. Each complete project will consist of three main components: (1) implementation, (2) experimentation, and (3) discussion. A single-person project should have a problem scope that is roughly twice the size of a homework assignment. A 2-person project should have a correspondingly larger scope.

To ensure that everyone is making progress, there will be graded milestones (described in more detail in later sections):

* Milestone 1: project plan [10 pts] (Due: Tue 4/17)
* Milestone 2: Project Checkpoint [15 pts] (Due: 5/1/18)
* Milestone 3: Project Presentation [15 pts] (Due: 5/14/18)
* Milestone 4: Project Code & Report [40 pts] (Due: 5/14/18)

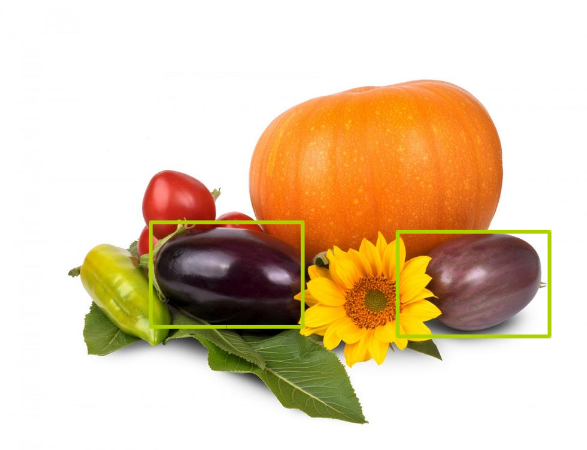
Coming up with an idea is often somewhat challenging. The main requirement is that your project should enable you to demonstrate that you’ve learned something about computer vision this semester. Below are a few ideas that may help.

* Apply techniques from class to a new problem:
  + Museum guide: given an image of a work of art, provide information about it.
  + Virtual fitting room: allow users to try on clothes virtually.
  + Shot prediction: predict if a basketball shot will go in based on the first few frames of video.
  + Tic-tac-toe: given an image of a tic-tac-toe board, automatically draw (on the image) the next best move. Point a webcam at a whiteboard and play interactively. (Or try [some other pencil-and-paper-based game](http://www.papg.com/).)
  + Leaf recognition: collect a dataset of leaf images and train a model for recognition.
  + Handwritten math solver: given an image of basic arithmetic (e.g., 2+5 = ), provide the correct answer.
  + Sign language: train a model to recognize (a subset of) standard sign language hand positions or gestures.
  + Tracking: given video of one or more moving objects (faces, people, cars, etc.), keep track of which is which over time.
  + [Turn a sketch into UI code](https://airbnb.design/sketching-interfaces/)
* Extend a homework or lab exercise (e.g., make an improved fruit finder or puzzle solver).
* Implement a topic we didn’t cover (e.g., augmented reality, [Seam carving for content-aware image resizing](http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.570.6321&rep=rep1&type=pdf), active contours, optical flow (or something else of interest to you).
* Implement from scratch an algorithm that we’ve used as a black box (e.g., a feature detector or descriptor).
* Find a project online and adapt it to a new or extended application.
  + [Face swapping](https://www.theverge.com/2018/2/11/16992986/fakeapp-deepfakes-ai-face-swapping)
  + [Splash of color](https://engineering.matterport.com/splash-of-color-instance-segmentation-with-mask-r-cnn-and-tensorflow-7c761e238b46)
  + [Google’s portrait mode](https://9to5google.com/2018/03/14/google-pixel-2-portrait-mode-open-source/)
  + [Image-to-image](https://affinelayer.com/pixsrv/index.html)
  + [Generating faces](https://zo7.github.io/blog/2016/09/25/generating-faces.html)
  + [Colorizing gray photos](https://blog.floydhub.com/colorizing-b&w-photos-with-neural-networks/?source=techstories.org)

## Milestone 1: Project Plan [10 pts] (Due: Tue 4/17)

Write up a document that describes a problem, your proposed solution, experiments you will run, and your evaluation criteria.

* (1 point) Problem: Describe the problem you are proposing to solve, why it is important, and what makes it difficult.
* (1 point) Solution: Describe your proposed solution at a high level, such as might be appropriate for a quick “elevator pitch.” (This will come in handy when you are trying to attract funding from venture capitalists for your new computer-vision-based start-up.)
* (2 points) Method: Describe your anticipated technical approach for the solution using computer vision terms. E.g., “I plan to use color thresholding to identify regions of the image corresponding to eggplants,” or “I plan to train a model for elbow detection using the Viola-Jones approach.” It’s fine to use techniques you don’t know yet, but read up enough to sound plausible. For example, if you are going to automatically predict whether a basketball shot will score or not, you should read up enough to be able to say “we will use least-squares estimation to find a best-fit parabola and determine if that parabola intersects with the basket or not,” even if you’re not sure how to implement that code yet. Make a mock-up example of the kind of output your algorithm should produce. E.g.,

* (1 point) Describe the data you plan to use. For example, you may capture your own images, download images from Flickr, write a script to scrape Google image search automatically, or download an existing computer vision dataset. (There are lots of datasets; try Google, or see, for example, [here](https://computervisiononline.com/datasets?page=1) or [here](http://riemenschneider.hayko.at/vision/dataset/). The [data science competition site, Kaggle, has lots of datasets](https://www.kaggle.com/datasets) as well; many of them are computer vision related.)
* (1 point) Describe the experiments you plan to perform to evaluate your algorithm. Discuss what kinds of metrics you will calculate (e.g., accuracy, precision, recall, time, etc.).
* (3 points) Describe grade targets.
* **Low Target.** This describes an amount of work which would be considered C-work. The expectation here is that your algorithm is at least functional, if only for a specific image/video.
* **Medium Target.** This is what would qualify as B-work. At this point, you would definitely need a working algorithm and at least some experimentation is performed.
* **High Target**. This list would include items which (if completed) would result in a high-quality algorithm (which could accept previously unseen input) and a robust set of experiments to demonstrate the effectiveness of your method. This list can also include “extras” which are unlikely to be completed in the course of the semester project, but may be interesting for potential future work.
* (1 point) Aiming for the high target, put together a short plan that breaks out your project by tasks (5 to 10 bullet points are fine) and your best estimate for how many hours will be required for each task. Include things like writing a script for data collection, background reading to come up to speed on a topic, writing your report, etc. Think about how long things have taken you on prior assignments to extrapolate reasonable estimates. Based on how many weeks are left in the semester, and how many hours you’ll devote each week, I would expect your estimate to be 30+ hours total.

This milestone will be graded on the reasonableness and completeness of this document. Points will be deducted for not describing in detail the experiments you will try and for overestimating the difficulty of the components. There are 3 possible outcomes to the grading of this milestone: (1) accepted as-is, (2) accepted with modification, and (3) sent back to the student to resubmit. Here are some tips for a successful first submission:

* You must specifically state what your low, medium, and high targets are.
* Do not use % accuracy as a target. This is very much dependent on your data and not the methods used. You could be either making things too easy or too hard for yourself.
* Describing your experiments does not mean how you're going to set up your system. You are to describe how you plan to evaluate and report how well your algorithm performs.
* It is not ok to just say you're going to use “some algorithm” or “things I learned in class.” You need to be clear about the specific approach you plan to take. Even if your chosen topic is something that we have yet to cover in class, you should familiarize yourself enough to write up a plausible proposal.
* When picking a project, think about which parts will be challenging. Not all aspects need to be difficult. For example, your project may incorporate relatively straightforward computer vision techniques, but require more effort developing a user-friendly interface or deploying on a mobile device. Or you might spend more time on data collection and experimentation and less on algorithm development.

## Milestone 2: Project Checkpoint [15 pts] (Due: 5/1/18)

For this milestone, you must submit your well-commented code in its current state and a status report. Your code should run and functionality should be at or approaching your low-target goals.

Your report should contain (1) overview of your project, (2) progress to date, (3) images showing current results from your code, (4) any roadblocks or issues thus far, and (5) your plan for finishing the remaining items. Submit one zip file in Moodle.

## Milestone 3: Project Presentation [15 pts] (Due: 5/14/18)

You will deliver an in-class presentation during the assigned final exam period. The presentation should be 5-7 minutes. Your grade for this milestone will be partially determined by your classmates, so you'll want to make sure that your presentation is clear, well-rehearsed, and describes your project effectively. Since we’re doing computer vision, you should aim to have lots of pictures and not too much text. You are welcome to attempt a live demo if desired.

Submit your slides (PowerPoint, PDF, or Google Slides) in Moodle by the start of the final exam period.

## Milestone 4: Project Code & Report [40 pts] (Due: 5/14/18)

Submit, in Moodle, a single zip file containing your python code and a written report describing the project (by the start of the final exam period). (Do not submit large datasets of images.) The report should be formatted as a PDF file. The rubric below shows the details of what should be included.

| Criteria  (weight) | 1  Great | 0.6  Ok | 0.2  Needs Improvement |
| --- | --- | --- | --- |
| Code functionality  (x10) | Code functionality substantially meets the high-target goals. | Code functionality substantially meets the medium-target goals. | Code functionality substantially meets the low-target goals. |
| Code style  (x2) | Code is clear, well-commented, broken up into reasonable-sized functions, makes use of constants and variables and avoids magic numbers, does not contain  Code is Python 3.5, easy to run, neatly organized and commented, , broken up into reasonable-sized functions, makes use of well-named constants and variables, includes attributed names/links for ANY AND ALL code from other sources, does not include extraneous or irrelevant code (e.g., copy-pasted from somewhere else). Code may be broken up into individual files. | Minor issues | Code is sloppy and hard to read |
| Intro /Problem  Statement  (x2) | Set up the problem, including background, why the problem is challenging, what your grade targets were, and which you have attempted. | A few details omitted. | Document fails to introduce the problem. |
| Method / Discussion of Process  (x8) | Document clearly describes the process followed so that another student in the class could implement the algorithm. Any hyperparameters, thresholds, and other such numbers should be discussed. Intermediate images showing how each major piece of functionality should be interwoven in the text. | Document describes the overall process followed, but omits a few details. | Document fails to describe the overall process followed. |
| Results  (x8) | Document includes qualitative results that show how the algorithm performs on selected examples. Both success and failure cases are shown. Quantitative results measuring algorithm performance (e.g., accuracy, precision, recall, etc.) are shown, as well as timing information. | Some results not reported. | Very few results reported. |
| Conclusions / Discussion  (x8) | A discussion of the difficulties of this problem, how you could measure the quality of your algorithm, why your algorithm did (or did not) work and how it could be improved. If there were any problems with your implementation (e.g. clearly wrong output) then make sure to indicate that in your write-up and provide as much information as you can as to what you think is causing the problem.  **Make sure to indicate which grade target you believe you have achieved.** | Same, but minor details missing. | Document doesn’t describe clear thoughts about results and future work. |
| Writing, organization, and aesthetics  (x2) | Clear, well-written, without grammatical issues. All images and tables nicely formatted, section headers used. | Minor issues | Document looks sloppy |