Agreement
This laboratory assignment will be completed in groups of two or three (max). Each team member must follow the team member guidelines developed by the students in this class, included in the next section. Instructor will reduce the grade of the team member who fails to follow the team guidelines generated in our AI group.

Lab Team Work Guidelines

1. Organization and function:
   - Team will delegate one member to be a team manager. In teams of two, the role of the team manager will be restricted to initiating the project on GitHub and resolving merge issues.
   - The project work will be distributed evenly among the team members. The work of being a team manager should count toward work distribution.
   - The team will work collaboratively to create a detailed project work plan with clear deadlines.
   - Teams must ensure that the project’s scope is feasible and that every member of the team is comfortable with the projected scope.

2. Communication:
   - All team members must promptly respond to the other team members’ communication.
   - All team members must regularly check and respond to Slack messages and GitHub issues.
   - Each team member should keep their communication clear and concise.

3. Interaction:
   - Conflicts within the team should be first brought to the team leader first. If unresolved, they should be elevated to the instructor.
   - All team members must recognize and avoid biases.
   - Each team member must act in a professional and in a respectful manner.
   - All team members must attend the scheduled team meetings and be punctual.

4. Project Problem Solving and Decision Making:
   - During problem solving teams should utilize collaborative brainstorming and researching tactics.
   - During brainstorming each team must ensure that every member is able to express their ideas, solutions, concerns, etc.
• The decisions about the project are to be made collaboratively. If a decision cannot be reached, a majority vote should be the deciding factor, or, in case of a draw, the instructor’s input should be the deciding vote.

Objectives

To learn how to use a computer vision OpenCV software and a Python programming language for computer vision applications. To correctly apply smoothing, blurring, thresholding and edge detection to images using OpenCV functions. To gain understanding of image processing techniques that can be used for specific computer vision applications. To apply a computer vision technique using supervised learning algorithm to a problem in baseball. To evaluate the performance of the developed system under various environmental conditions and to reflect on its design and development.

Reading Assignment

To learn more about OpenCV and various learning algorithms, you should carefully read and follow the chapters from the supplemental reading. You are also invited to study the provided example programs that are based on the projects by “pyimagesearch” and Allegheny students.

If you are not familiar with a Python programming language, you may refer to Python library documentation at https://docs.python.org/3/.

To learn more about applications of AI in baseball, you should read popular articles, such as What Major League Baseball Can Teach Us About AI in Business.

If you have not done so already, please read all of the relevant “GitHub Guides”, available at https://guides.github.com/, that explain how to use many of the features that GitHub provides. In particular, please make sure that you have read guides such as “Mastering Markdown” and “Documenting Your Projects on GitHub”; each of them will help you to understand how to use both GitHub and GitHub Classroom.

Accessing the Laboratory Assignment on GitHub

Since this is a team-based assignment we will be using a group assignment functionality of GitHub Classroom. For team assignments only one person will be creating the team while the other team members will join that team. Please form a team consisting of two or three members, assign one person to be the designated team manager.

The selected team manager should go into the #labs channel in our Slack team and find the announcement that provides a link for it. Copy this link and paste it into your web browser. Now, you should accept the laboratory assignment and create a new team with a unique and descriptive team name (under “Or Create a new team”).

Now the other members of the team can click on the assignment link in the #labs channel and select their team from the list under “Join an Existing Team”. When other team members join their group in GitHub Classroom, a team is created in our GitHub organization. Teams have pretty cool functionality, including threaded comments and emoji support. Every team member will be able to push and pull to their team’s repository. Your team’s project manager should be the one to resolve any conflicts or merge pull requests.

Unless you provide the instructor with documentation of the extenuating circumstances that you are facing, not working in a team and not accepting the assignment means that you automatically receive a failing grade for it.

Due: 28 February, 2020 at 2:30 pm
Baseball Project

As applications of AI in sports are becoming limitless, from recruiting efforts to training to marketing and fan interaction, various AI technologies have been emerging to assist with these efforts.

In this three-week lab students are invited to implement a learning agent using an open-source learning platform, OpenCV, for a baseball application. **Design choice:** You can select to complete one of the three applications of machine learning in baseball. You can read more about the details of the requirements of each application in the subsections 2.1 – 2.3 of this assignment sheet.

1. **Identifying and Tracking a Baseball:** use learning to detect and track a baseball and collect some data corresponding to it, such velocity, location of the ball, etc.

2. **Baseball Video Overlays and Tracking:** use OpenCV to create overlays with multiple videos of pitches and use learning to detect and output the ball at its start and end location.

3. **Learning the Kinematic Sequence of the Pitcher:** use learning to identify important parts of the body and track and record the changes in those as the pitch is being thrown.

---

1. Image Manipulation and Transformation

Before you immerse into an implementation of a selected application, you will first experiment with OpenCV and Python by going through the given program and following the tutorial style book given as a supplement. If you are not familiar with a Python programming language, you may refer to Python library documentation. You will also find additional introductory materials on Python in the lab3 directory of the lab repository.

1. Study the program called `src/colorspaces.py` in the ‘src1’ directory of the lab repository. Please refer to the OpenCV Color Conversation Documentation and read Chapter 6.6 from the “Practical Python and OpenCV” book to enhance your understanding of the program.

2. Follow the Chapters 8, 9 and 10 in the “Practical Python and OpenCV” book to learn about smoothing and blurring, thresholding, and edge detection. As a result of this work, you should produce the following programs, saved into the src1 directory: `blurring.py`, `simple_thresholding.py`, `adaptive_thresholding.py`, `otsu_and_riddler.py`, `sobel_and_laplacian.py`, and `canny.py`. If you are using Docker Desktop, please remember that you will need to modify the program in the book so that it works in the docker container by replacing `imshow` method with `imwrite` method.

3. Apply the edge detection program to your own image. You may need to modify the program to ensure it is able to properly detect object edges in your image. Save your output.

This part of the lab is due by 2:30pm on February 14. Once you have gained more experience using OpenCV and learning algorithms, you can move on to the main component of the lab, see details below. By February 14 deadline, you must research the given applications, choose one to pursue, and document your selection in an appropriate section of the report.

2.1 Baseball Detection and Tracking

In this problem, you are to use OpenCV, Python programming, and input video files to create a baseball detection and tracking system. The goal of this project is to detect a ball (in the glove of the pitcher) and track the ball as it travels, collecting various statistics about it throughout its travel. You must gather at least two of the following data points:
speed of the ball (distance/time),
spin axis,
spin rate,
horizontal/vertical break,
location of the ball.

The data must be collected when the ball is released and caught (or missed) and at various times during its travel.

There are many algorithms in OpenCV for performing object detection, tracking, and analysis. For object detection you can use a supervised learning algorithm, a classifier, that uses certain features of the image to correctly label them as a specific object or not. **Design choice:** you need to select a learning algorithm that you will utilize (e.g., Cascade, SVM, etc.) and a feature descriptor that your algorithm will use (e.g., Haar, HOG, LBP). I suggest using LBP or HOG features, as they are integer, fewer and more discriminant in contrast to Haar features, so both training and detection with LBP and HOG are several times faster then with Haar features. Although, you can collect your own labeled data (images) and then train your classifier with those, that task is time intensive. To save time, first look for a classifier that either has already been trained using specific data sets and a certain feature descriptor in OpenCV or select a feature descriptor that already has compiled labeled data set that can be used for training. You can utilize class programs, existing open source projects, and OpenCV documentation for usage examples of various methods. You are also free to select an algorithm not discussed in class.

### 2.2 Baseball Video Overlay and Baseball Detection

In this problem, you are to use OpenCV, Python programming, and input video files to create an overlay of at least three pitching videos. The goal of this project is to create overlay to synthesize various types of pitches and to record the location of the ball at the start and the final destination. To see examples of baseball video overlays, you can view the following videos:

- [https://www.youtube.com/watch?v=1EW7YFGegTg](https://www.youtube.com/watch?v=1EW7YFGegTg)
- [https://www.youtube.com/watch?v=jUbAAurrnwU](https://www.youtube.com/watch?v=jUbAAurrnwU)

The images identifying the balls in overlayed videos as they begin travel and they reach their target must be saved.

To make an overlay you can apply the `addWeighted` function in OpenCV. Then, you will need to add an object detection algorithm (as in the project 2.1) to detect the balls. For this task, you can use a supervised learning algorithm, a classifier, that uses certain features of the image to correctly label them as a specific object or not. **Design choice:** you need to select a learning algorithm that you will utilize (e.g., Cascade, SVM, etc.) and a feature descriptor that your algorithm will use (e.g., Haar, HOG, LBP). You can utilize class programs, existing open source projects, and OpenCV documentation for usage examples of various methods. You are also free to select an algorithm not discussed in class.
2.3 Kinematic Sequence Learning

In this problem, you are to use OpenCV, Python programming, and input video files to create a system that outputs various components of the kinematic sequence. To learn about the kinematic system, you can read an article titled “Understanding the Kinematic Sequence”, and follow through with the related articles at the bottom of that page. You must detect and track at least two of the following body movements:

- pelvis,
- chest,
- upper arm,
- forearm/wrist.

This is the most challenging of the three projects as accurate detection and tracking of body components typically requires a method called “pose estimation” and “skeletonization”. The successful existing implementations of these methods involve using deep learning algorithms with neural networks (https://www.learnopencv.com/tag/human-pose-estimation/), a topic we will not cover for a few more weeks. However, you may implement basic functionality of kinematic sequencing, although perhaps not very accurately, using standard OpenCV learning algorithms.

3. Experimental Analysis

After you have conducted preliminary tests to ensure your program(s) runs without errors and you are satisfied with your detection accuracy, you need to conduct environment comparison experiments. Specifically, we are interested in learning the performance of object detection and tracking algorithms for different camera positions. For these experiments, you can use the given videos or obtain your own and produce performance measure metrics for the different camera angles. You must evaluate at least two different camera angles.

Testing your Program

Since this is an open-ended assignment involving a method of your choice you must execute your program(s) and see if they are producing the expected output. When you believe you have completed all of the requirements of the writing for this lab, you can use the GatorGrader tool (https://github.com/GatorEducator/gatorgrader) to automatically check if your submission meets the minimum submission requirements (by running the “gradle grade” command). If you are using Docker Desktop, you can use the following “docker run” command to start “gradle grade” as a containerized application, using the “DockaGator” Docker image available on DockerHub. You can run the following command to run the “gradle grade” on your project:

docker run --rm --name dockagator \
  -v "$(pwd)="/project \
  -v "$HOME/.dockagator="/root/.local/share \
  gatoreducator/dockagator

This command will use "$(pwd)" (i.e., the current directory) as the project directory and "$HOME/.dockagator" as the cached GatorGrader directory. Please note that both of these directories must exist, although only the project directory must contain some content. To ensure that the previous
command will work correctly, you should create the cache directory by running the command “mkdir
$HOME/.dockagator”; you will only need to do this once. If the above “docker run” command does
not work correctly on the Windows operating system, then you may need to instead run the
following command to work around limitations in the terminal window:

docker run --rm --name dockagator \
   -v "$(pwd)":/project \
   -v "$HOME/.dockagator:/root/.local/share" \
   gatoreducator/dockagator

To enter into an “interactive terminal” in the Docker container, you can instead use the following
command

docker run -it --rm --name dockagator \
   -v "$(pwd)":/project \
   -v "$HOME/.dockagator:/root/.local/share" \
   gatoreducator/dockagator /bin/bash

Now, you can type “gradle grade” to run through the checks that the instructor specified for your
assignment.

Throughout the process of working on this assignment you should remember to transfer your files
to GitHub using the “git commit” and “git push” commands. Remember, to correctly complete
this assignment you need to commit all modified files to GitHub. Also, please utilize appropriate
GitHub practices while working with your team members to avoid common merge conflicts.

When you use the “git push” command to transfer your source code to your GitHub repository,
Travis CI will initialize a “build” of your assignment, checking to see if it meets all of the minimum
requirements. If your commit meets all of the minimum submission established requirements, then
you will see a green ✔ in the listing of commits in GitHub after awhile. If your submission does not
meet the requirements, a red ✗ will appear instead. Please note GatorGrader and Travis are used
to just ensure that all materials have been submitted. A Travis pass does not indicate a correct
completion of the technical details of the lab due to the open ended nature of the lab.

Project Walkthrough

During the lab session on Friday, February 21st, each team will participate in the project
walkthrough process. Project walkthrough is an informal process where the instructor leads the
process of reviewing the progress of the project and the written code is reviewed for technical
accuracy with the objective of finding errors and improving the quality of the code. The purpose
of this walkthrough is to motivate continuous progression on the project, identification of any
conceptual issues, and detection of any technical errors. When the walkthrough is finished, the
authors of the project are responsible for taking the necessary actions to correct the identified
issues.

By this project walkthrough, each team should have a tentative working prototype of their
solution. It does not need to be complete or work accurately but it should accomplish most of
the goals. During the walkthrough, the team members will collaboratively lead the walkthrough
process, which should last 5-10 minutes for each team. Each team should:

• Identify the chosen application and describe the developed conceptual solution.

• Discuss the learning agent components (algorithms utilized).

• Explain the written code.

• Identify the steps left to complete this project.
Project Demonstration

During the lab session on Friday, February 28th, each team will be given an opportunity to demonstrate their project. When the lab session starts, teams will be given a few minutes to organize their demonstrations. Then, taking turns, each team will showcase their results in front of the class. During this five minute demo, each team should address the following (you do not need to prepare slides):

- Introduction of the team.
- The overview of the project completed.
- The demonstration.
- Summary of the results.

Finally, class members will be invited to participate in an interactive demonstration session, where the instructor will assess the technical knowledge of the developed system, and everyone will be able to dive into each other’s demonstration in more detail.

Required Deliverables

This assignment invites you to submit the following deliverables through your team repository.

1. Part 1 of the lab (src/blurring.py, src/simple_thresholding.py, src/adaptive_thresholding.py, src/otsu_and_riddler.py, src/sobel_and_laplacian.py, and src/canny.py programs and the produced images) and the project selection are due on February 14 by 2:30pm. The rest of the written requirements are due at 2:30 pm on February 28.

2. A properly completed and commented source program(s) for part 2 of the lab.

3. The report, stored in /writing/report.md and written in Markdown, that contains answers to all sections in the report including experimental analysis(follow the prompts inside the report document).

4. Lab session on February 14 will be used for lab work.

5. Lab session on February 21 will be used to conduct project walkthroughs and for lab work.

6. The lab session on February 28 will be used for demonstrations.

7. Revisions to the lab submission will be accepted for partial credit until March 6.

Evaluation of Your Laboratory Assignment

The grade that a student receives through Sakai on this assignment will have the following components:

- **Percentage of Correct GatorGrader Checks and Travis CI Build Status [up to 10%]:** Students are encouraged to repeatedly revise their submission to ensure that it passes all of GatorGrader’s checks about, for instance, the length of the report, its appropriate use of Markdown, and inclusion of minimal programming constraints. The instructor will provide GatorGrader and Travis configuration files during the second week of the lab to each team individually that will be customized to their work.
• **Mastery of Verbal Explanation during Walkthrough and Demonstration [up to 25%]:** Since the continuous and timely project development and the ability to communicate technical details of a project is crucial to building successful software and hardware applications, a portion of students' lab grade will be determined based on the quality of the project walkthrough and the project demonstration.

• **Mastery of Technical Writing [up to 10%]:** Students will also receive a portion of the lab grade when the responses to the technical writing questions presented in the ‘writing/report.md’ reveal a mastery of both writing skills and conceptual and technical knowledge. To receive this portion of the grade, the submitted writing should have correct spelling, grammar, and punctuation in addition to following the rules of Markdown and providing conceptually and technically accurate answers.

• **Mastery of Technical Knowledge and Skills [up to 55%]:** Students will receive a portion of their assignment grade when their project implementation reveals that they have mastered all of the technical knowledge and skills developed during the completion of this project. As a part of this grade, the instructor will assess aspects of the project including, but not limited to, the correctness of the program, the completeness and correctness of the software and hardware integration, the use of effective source code comments and Git commit messages.