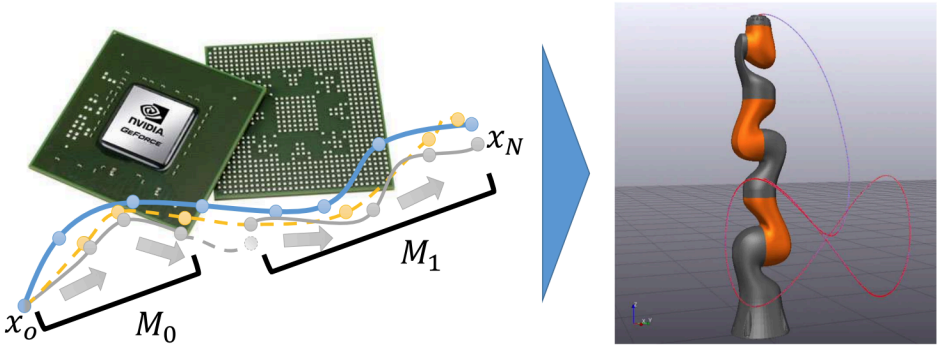
COMS BC3159: Parallel Optimization for Robotics



Semester: Fall 2024 | **Instructor:** Brian Plancher
Credit: 3 points | **Meeting time:** MW 1:10pm-2:25pm | **Room:** Milbank 405
Prerequisites: COMS W3251 Computational Linear Algebra, MATH UN1201 Calculus III, and COMS W3157 Advanced Programming or CSEE W3827 Fundamentals of Computer Systems or Prior Experience with C(++) Programming (e.g., pointers, arrays, and memory management). Contact the instructor if you have equivalent prior experience but do not have prerequisites.
**Enrollment Capped at 75 Students (Instructor Managed Wait List)*

Description:

Many stages of state-of-the-art robotics pipelines rely on the solutions of underlying optimization algorithms. Unfortunately, many of these approaches rely on simplifications and conservative approximations in order to reduce their computational complexity and support online operation. At the same time, parallelism has been used to significantly increase the throughput of computationally expensive algorithms across the field of computer science. And, with the widespread adoption of parallel computing platforms such as GPUs, it is natural to consider whether these architectures can benefit robotics researchers interested in solving computationally constrained problems online. This course will provide students with an introduction to both parallel programming on GPUs as well as numerical optimization. It will then dive into the intersection of those fields through case studies of recent state-of-the-art research and culminate in a team-based final project.

Learning Outcomes:

By the end of the semester, you will be able to:

- Understand the opportunities and limitations of parallel programming on GPUs
- Understand the opportunities and challenges of numerical optimization algorithms
- Engage critically with recent research on parallel optimization algorithms for robotics
- Collaborate with a team to develop and present an open-ended final project

Grading:

- 35% Problem Sets
- 30% Exam
- 30% Final Project Presentation, Report, and other Milestones
- 5% Attendance, Collaboration, and Participation

Note: requests for regrades can only be made for 1 week following the return of a grade.

Student drop-ins (Office Hours)

See brianplancher.com/office_hours for the most up-to-date schedule!

- These are scheduled time-slots for us to chat and I will also be available by appointment in case the scheduled slot doesn't work for you!
- I strongly encourage you to come to at least one student drop-in slot per semester.
- This is your time, so you can use it however you want! Stop by for a quick chat, ask questions about the course (or give feedback), ask questions about research, ask questions about getting jobs in finance/consulting/tech, tell me about a new hit TV show, any reason is a good reason!

Course Outline:

A high-level course outline follows. View the Course Schedule below for more information.

Module 0: Introduction and Course Overview

This module lays the foundation for the course. Namely, it provides an introduction to both optimization algorithms commonly used for robotics applications as well as the opportunities for parallel programming.

Module 1: (GPU) Parallel Programming

This module provides an introduction to parallel programming with a focus on GPUs. We will cover both common software constructs as well as practical implications of hardware architectures on programming models and optimizations.

Module 2: Optimization Algorithms for Robotics

This module provides an introduction to numerical optimization with a focus on algorithms used for robotics applications. We will cover both theoretical mathematical foundations and practical application tradeoffs, constraints, and limitations.

Module 3: Putting it All Together

This module integrates Modules 1 and 2 exploring how recent robotics research has leveraged parallelism to accelerate optimization algorithms for robotics.

Module 4: Final Integrative Projects

Throughout the course we will develop an integrative, team-based, final project which will be presented during the last class.

Readings and Materials:

The course will involve the reading of technical papers, chapters of textbooks, and technical blog posts and the watching of technical videos. Specific readings and videos are listed in the detailed reading list and course scheduled below. Readings and videos can be accessed for free either as they are either open-source or will be posted to Canvas. Students will not need to purchase any textbooks or other materials for this course. In particular, all GPU programming will leverage cloud-based platforms to ensure that no students need to purchase additional computer hardware.

Email and Slack Policy:

I request that as much as possible you use Slack (instead of email) for all course-related questions. Please post liberally as if you have a question, another student likely has a similar question. There is also an anonymous bot if you would like to submit anonymously. I will try to respond to all Slack posts as soon as possible during working hours and at least within 1 business day during the weekdays and within 2 business days over the weekend. **Click this link to sign up for our course Slack**. If you do need to reach me via email, please send the email to bplancher+courses@barnard.edu so that it is routed appropriately. I will try to respond to all emails within 2 business days.

Gradescope and Regrade Policy:

We will be using Gradescope for problem set submission and autograding. All coding assignments with autograders can be submitted an infinite amount of times until the deadline. As such, re-grades on such assignments will only be allowed in extenuating circumstances. Other assignments will be manually graded following submission. All re-grade requests for such assignments must be submitted as a private message on Slack and will only be accepted for one week following the return of a grade. Do note that grades could go up or down following a re-grade.

Late Policy:

The late policy of this class follows a policy found in many Barnard/Columbia CS courses. Each day (24-hour period) or partial day late incurs a 25% penalty on the assignment. However, you are allowed a total of 5 “flex” days, to be used as you wish throughout the semester, with a maximum of 2 per assignment. Late hours round up the nearest day. Note that these are NOT applicable on any final project related deadlines. To use a “flex” day, simply submit your work late and send a note to the course staff indicating how many “flex” days your late submission has incurred, and how many remaining “flex” days you have. When possible, advance notice is appreciated. If there is a situation that you feel should be exempt from this policy, you must reach out over email at least 48-hours prior to the due date.

Academic Integrity:

You are expected to hold yourself to the highest standard of academic integrity and honesty, as reflected in the Barnard Honor Code. Approved by the student body in 1912 and updated in 2016, the Code states:

“We, the students of Barnard College, resolve to uphold the honor of the College by engaging with integrity in all of our academic pursuits. We affirm that academic integrity is the honorable creation and presentation of our own work. We acknowledge that it is our responsibility to seek clarification of proper forms of collaboration and use of academic resources in all assignments or exams. We consider academic integrity to include the proper use and care for all print, electronic, or other academic resources. We will respect the rights of others to engage in pursuit of learning in order to uphold our commitment to honor. We pledge to do all that is in our power to create a spirit of honesty and honor for its own sake.”

This course’s policy on academic honesty builds on the honor code and is best stated as "be reasonable." We recognize that interactions with classmates and others can facilitate mastery of the course’s material. As this course revolves mostly around team projects we expect students to collaborate heavily and work together on those assignments. Even on individual assignments students should feel encouraged to ask classmates and others for conceptual help. However, there remains a line between asking for help and submitting someone else’s work. Especially on individual assignments, make sure this collaboration does not reduce to your classmate doing your work for you (e.g., writing your response, copy-pasting code, or making your slides). If in doubt as to whether some act is reasonable, ask first! The course staff would

much rather have a conversation about extensions than about academic integrity! We hope you are reading these policies (or at least skimming them), so if you are, please send the course instructor an email with a (robotics) pun/joke/meme with the subject “academic integrity easter egg” and if you come by office hours I’ll give you a prize (exact prize subject to availability)! Acts considered not reasonable will be referred to the Barnard Honor Board, and the course reserves the right to impose local sanctions on top of that outcome. **If you commit some act that is not reasonable but bring it to the attention of the course staff within 48 hours, the course may impose local sanctions, but the course will not refer the matter further except in cases of repeated acts.**

Diversity, Inclusion, and Accessibility:

In an ideal world, science would be objective. However, much of science is subjective and is historically built on a small subset of privileged voices. We acknowledge that it is possible that there may be both overt and covert biases in the material due to the lens with which it was written, even though the material is primarily of a scientific nature. Since integrating a diverse set of experiences is important for a more comprehensive understanding of science please contact the course staff (in person or electronically) or submit anonymous feedback if you have any suggestions to improve the quality of the course materials. We would like to create a learning environment that supports diversity of thoughts, perspectives, and experiences, and honors your identities. If you have a name and/or set of pronouns that differ from those that appear in your official records, please let us know! If you feel like your performance in the class is being impacted by your experiences outside of class, please don’t hesitate to contact us. If you prefer to speak with someone outside of the course, the Center for Engaged Pedagogy (CEP) or the office of vice president for Inclusion and Engaged Learning are excellent resources.

If you believe you may encounter barriers to the academic environment due to a documented disability or emerging health challenges, please contact the course staff or the Center for Accessibility Resources & Disability Services (CARDS). So that the course staff has enough time to implement accommodations, we request that any student with approved academic accommodations contacts the course staff within the first three weeks of the semester. If you have questions regarding registering a disability or receiving accommodations for the semester, please contact CARDS at (212) 854-4634, cards@barnard.edu, or learn more at barnard.edu/disabilityservices. CARDS is located in 101 Altschul Hall.

Wellness Statement:

It is important for undergraduates to recognize and identify the different pressures, burdens, and stressors you may be facing, whether personal, emotional, physical, financial, mental, or academic. We as a community urge you to make yourself– your own health, sanity, and wellness–your priority throughout this term and your career here. Sleep, exercise, and eating well can all be a part of a healthy regimen to cope with stress. Resources exist to support you in several sectors of your life, and we encourage you to make use of them. Should you have any questions about navigating these resources, please visit these sites:

- barnard.edu/primarycare
- barnard.edu/about-counseling
- barnard.edu/wellwoman/about
- health.columbia.edu/stressbuster

Affordable Access to Course Texts & Materials:

All students deserve to be able to study and make use of course texts and materials regardless of cost. Barnard librarians have partnered with students, faculty, and staff to find ways to increase student access to textbooks. By the first day of advance registration for each term, faculty will have provided information

about required texts for each course on CourseWorks (including ISBN or author, title, publisher, copyright date, and price), which can be viewed by students. A number of costfree or low-cost methods for accessing some types of course texts are detailed in the Barnard Library Textbook Affordability guide (https://library.barnard.edu/textbook-affordability). Undergraduate students who identify as first-generation and/or low income students may check out items from the FLIP lending libraries in the Barnard Library (library.barnard.edu/flip) and in Butler Library for an entire semester. Students may also consult with their professors, the Dean of Studies, and the Financial Aid Oce about additional affordable alternatives for having access to course texts. Visit the guide and talk to your professors and librarian for more details.

Use of AI Content Generators:

I view AI tools as a powerful resource that you will likely leverage in the future. As such, the use of AI-based content generation tools, such as ChatGPT, is permitted in this course. However, you will be required to disclose any use of AI tools for each assignment.

The goal of this policy is to help you develop your resilience to automation, as these tools will become increasingly prevalent in the future, and also to learn about their weaknesses. By incorporating these tools into your work process, you will be able to focus on skills that will remain relevant despite the rise of automation. However, **it is important to note that AI tools are susceptible to errors** (e.g., most citations are incorrect). As a student, it is your responsibility to ensure the quality and appropriateness of the work you submit in this course. As such please make sure to read carefully (and likely heavily edit) the output from such tools. Also, please be mindful of the data you provide to these systems, as your work may contain private information, not just your own but also that of others. For example, you should never enter the names of study participants into ChatGPT. Furthermore, there is a risk of inadvertently plagiarizing when using these tools as they often draw content without proper citation. Standard plagiarism policies will apply to all assignment submissions, and “AI did it!” is not a sufficient excuse. To prevent this, you can consider using more responsible tools that are designed to cite their data sources, such as Perplexity AI, and in either case you should make sure to add citations where appropriate yourself. Lastly, be aware of the dangers of becoming overly dependent on these tools. While they can be incredibly useful, relying on them too much can diminish your own critical thinking and writing skills.

If you do not wish to use these tools, that is a valid decision. The use of AI tools in education can be messy and unpredictable due to the risks mentioned earlier. You may have moral confusion or concerns about the uncertainty associated with using AI tools in their coursework. If you do not wish to use them, that is a valid decision. This policy aims to anticipate and mitigate any potential harms associated with AI tool usage, rather than promoting their use.

As mentioned above, for every assignment submission, you are required to include an “AI Tool Disclosure” paragraph which states to what extent you used AI tools. We will not mark you down for the use or non-use of AI tools. The course staff simply wants to understand the prevalence of AI tool use and methods of use to better adapt course policies and teaching practices for the future.

Assignment Descriptions:

Problem Sets (35%)

Purpose:

- Develop parallel programming skills to prepare for the final project and future courses, research, and industry jobs.
- Gain confidence with the mathematical skills that power optimization algorithms.

Students will work through a series of problems in order to develop parallel programming and optimization skills. These problem sets will include both coding and theoretical components. Students will learn how to use version control software and will also submit technical solutions through autograders and as written PDFs. This will both help prepare students for final projects and help develop skills they can use in future courses and jobs. Each problem set will be paired with an in-class "party" to aid in collaborative problem solving although students must submit their own answers.

Exam (30%)

Purpose:

- To reinforce and validate knowledge learned across both optimization and parallel programming including both theoretical foundations and practical applications for use on the final project, and future courses, research, and industry jobs.

The exam will be held in-class, closed-book, no notes. It will cover all of the course material presented in both the optimization and parallel programming modules of the course. See the schedule for dates and topics covered. A review will be held preceding the exam. The exam will be designed to test knowledge useful for future work and validate your knowledge of both the core mathematical and computational foundations explored in this course and their applications.

Final Project Presentation, Report, and other Milestones (30%)

Writeup Purpose:

- Develop the skill of developing a team-based project report
- Get feedback on technical writing skills (and practice using LaTeX)

Presentation Purpose:

- Develop the skill of presenting a team-based project to an audience
- Get feedback on presentation skills both in terms of delivery and slides

Other Milestones Purpose:

- Develop a project proposal and get feedback early and often on a project design and plan

Students will work in small teams of 2-3 students (special requests must be made for smaller or larger groups) on a final integrative project and each team will be asked to provide a technical report on and present their final project. The report will be in the form of a standard academic publication. A few resources on writing academic papers can be found here, here, and here. We will use LaTeX for writing the report as it is standard practice in computer science. Overleaf is an online editor (think Google Docs) that allows you to write LaTeX and never have to work on package management, installation, setup etc. They also have a great guide for getting started (and the course staff will upload a skeleton project you can copy with all of the necessary templates). The presentation should be supported by slides. Two guides for good slide design can be found here and also here. Also, here is a guide on effective presentations. More details on the final project will be distributed later in the semester.

Attendance, Collaboration, and Participation (5%)

Purpose:

- To develop collaboration and teamwork skills

This course is capped at 75 students which means it will be a moderately sized course with many interactive "lab-style" classes. As such you are expected to attend lectures and participate in class discussions. Similarly as the final project will be done in collaboration with other students, collaboration and group participation will be key skills you will need to work on during this class. This is intentional as most of your future work environments will rely heavily on teamwork and team based projects and presentations whether you go into technology or business in industry or academia.

Preliminary Detailed Reading List:

As mentioned above, this course will involve the reading of technical papers, chapters of textbooks, and technical blog posts and the watching of technical videos. Specific readings and videos are listed in the detailed course scheduled below. Readings and videos can be accessed for free either as they are open-source or through the school library or will be posted to Canvas. Students will not need to purchase any textbooks or materials for this course.

Textbooks

- Stuart J. Russell. *Artificial Intelligence a Modern Approach*. Pearson Education, Inc., 2010.
- Stephen P. Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.
- Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, 1999.
- Tedrake, Russ. *Underactuated Robotics*. Online, 2022.
- Michael S Kirkpatrick. "OpenCSF: An Online Interactive Textbook for Computer Systems Fundamentals." *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*. 2018.
- Matthews, Suzanne J., Tia Newhall, and Kevin C. Webb. *Dive Into Systems: A Gentle Introduction to Computer Systems*. No Starch Press, 2022.
- David B. Kirk and W. Hwu Wen-Mei. *Programming Massively Parallel Processors: a Hands-On Approach*. Morgan Kaufmann, 2016.

Peer-Reviewed Papers

- Hadi Esmaeilzadeh, Emily Blem, Renée St. Amant, Karthikeyan Sankaralingam, and Doug Burger. "Dark Silicon and the End of Multicore Scaling." *International Symposium on Computer Architecture*. 2011.
- Brian Plancher and Scott Kuindersma. "A Performance Analysis of Parallel Differential Dynamic Programming on a GPU." *International Workshop on the Algorithmic Foundations of Robotics*. Springer, Cham, 2018.
- Grady Williams, Andrew Aldrich, and Evangelos A. Theodorou. "Model Predictive Path Integral Control: From Theory to Parallel Computation." *Journal of Guidance, Control, and Dynamics* 40.2. 2017.
- Brian Plancher, Sabrina M. Neuman, Radhika Ghosal, Scott Kuindersma, and Vijay Janapa Reddi. "GRiD: GPU-Accelerated Rigid Body Dynamics with Analytical Gradients." *International Conference on Robotics and Automation*. 2022.

Other Resources

- Garrett Thomas. "Mathematics for Machine Learning." University of California, Berkeley (2018).
- Nick White. “Git Tutorial For Dummies.” 2021. https://www.youtube.com/watch?v=mJ-qvsxPHpY
- Mike Bailey. “Parallel Programming: Speedups and Amdahl’s law.” 2021. https://web.engr.oregonstate.edu/~mjb/cs575/Handouts/speedups.and.amdahls.law.1pp.pdf

Preliminary Course Schedule:

Week	Day	Date	Topic	Description	Assignments	Readings	Module
0	W	Sep 4	Intro Class	Overview of the Course, Nuts and Bolts, Optimization in Robotics, Thinking in Parallel	PS 0 Released (W): Math and Coding Background and Environment	Intro to Git [1,2] Math Reference Dive Into Systems Ch 1,2	Intro
1	M	Sep 9	Computer Systems, and Parallel Programming Concepts	Review on Computer Systems and Memory Hierarchy, Basic Parallelism including: Threads and Synchronization		Dark Silicon Dive Into Systems Ch 11 OpenCSF Ch 6.1-6.3, 7.1-7.2	(GPU) Parallel Programming
1	W	Sep 11	GPU Parallelism and CUDA	The GPU Computational Model (Blocks, Threads, SMs), NVCC, Host, Device, Synchronization, Shared, Global, Local, Host, and Unified Memory, I/O, Reductions			
2	M	Sep 16			PS 0 Due (M) PS 1 Released (M): GPU Parallel Programming	Dive Into Systems Ch 14,15 Kirk Ch 2, 3.1-3.2, 4.1-4.5, 5.1, 6.1	
2	W	Sep 18					
3	M	Sep 23	PS1 "Party"				
3	W	Sep 25	Advanced GPU Topics	TBD			
4	M	Sep 30	Convex Optimization and Vector Calculus	Convexity, Global vs. Local Optima, Gradient Descent, Vector Calculus	PS 1 Due (M) PS 2 Released (M): Optimization	Boyd Ch 1.1-1.4, 2.1, 3.1-3.1.4	Optimization Algorithms for Robotics
4	W	Oct 2	Nonlinear and Constrained Optimization	Taylor Expansions, Line Searches, Trust Regions, Penalty Methods, (Augmented) Lagrangian, KKT System		Nocedal Ch 2 Underactuated Ch 1.1-1.6, 2	
5	M	Oct 7	PS2 "Party"				
5	W	Oct 9	Numerical Optimization in	The Trajopt Problem, Direct Transcription and Colocation, Solving		AIMA Ch 17.1-17.3 Underactuated Ch	

			Robotics	Linear Systems, iLQR, and DDP and its ties to Bellman, Dynamic Programming		7.1-7.3, 8.1-8.2, 10-10.4	
6	M	Oct 14	Trajectory Optimization I		PS2 Due (M) PS 3 Released (M): Trajectory Optimization		
6	W	Oct 16	Trajectory Optimization II				
7	M	Oct 21	PS3 "Party"				
7	W	Oct 23	Advanced Optimization Topics	TBD	PS 3 Due (F)		
8	M	Oct 28	Course Review				Putting it All Together
8	W	Oct 30	Practice Exam Questions "Party"				
9	M	Nov 4	Election Day Holiday				
9	W	Nov 6	Exam				
10	M	Nov 11	Putting it All Together: Parallelism in Robotics Optimization	Project Overview, Parallel DDP, GRiD and Code-Generation, Instruction vs. Algorithm Level Parallelism	Project Instructions Released (W)	MPPI Paper GRiD Paper PDDP Paper	
10	W	Nov 13					
11	M	Nov 18	Project Proposal Meetings		Project Proposal Due (Su)		Final Project
11	W	Nov 20	Final Project Lab Time				
12	M	Nov 25					
12	W	Nov 27	Thanksgiving Holiday				
13	M	Dec 2	Project Update Meetings		Project Update Due (Su)		
13	W	Dec 4	Final Project Lab Time				
14	M	Dec 9	Final Project Presentations		Project Report Due (M)		