Seminar Topic
Computing Motion: Search, Planning, Control and Learning

Class Info
Meeting Time: M-W 3:20 - 4:40 pm
Classroom: CBIM 22
Credits: 3 (Counts as a Category B course for CS Graduate Students)
Webpage: `http://www.pracsyslab.org/cs673`

Check the above webpage for updated syllabus information and an online calendar of covered material. A Sakai website is also available and provides related resources.

Instructor
Kostas Bekris, Assistant Professor
CBIM, Computer Science Department

Office: CBIM 9
Hours: M-W 1:30 - 2:30 pm or after coordination via email
Email: kostas.bekris/AT/cs.rutgers.edu
Phone: 848 445-8854
Webpage: `http://www.pracsyslab.org/bekris`

Description
Computing motion in an automated way is useful in many application domains, including robotics, manufacturing, medical devices, transportation challenges, cyber-physical systems, mobile computing, mobile sensor networks, computer simulation and animation, as well as computer games.

Given the importance of this problem, different research communities have approached it from different perspectives and have proposed alternatives methodologies for addressing such challenges. Examples include heuristic search, algorithmic motion planning, optimal control and reinforcement learning. Due to both technological and algorithmic progress, these fields, which were rather distinct in the past, are presently on a collision course. The focal point of this intersection is the development of efficient algorithms for computing motion in an automated and efficient way.

In robotics, the problem originally involved moving a free flying rigid body without collisions (e.g., think of moving a piano from one room to another in a house without hitting the walls). The field has grown, however, to include complications such as uncertainty, multiple bodies, and dynamics. In AI, planning originally meant a search for a sequence of actions that transform an initial state into a desired goal state, which can often be approached by heuristic search methods. Presently, planning extends beyond this to include many decision-theoretic ideas such as Markov Decision Processes, imperfect state information, and game-theory. Furthermore, reinforcement learning is increasingly gaining ground as the way to address challenges where the underlying model of the moving system is uncertain or inaccurate. Although control theory has traditionally been concerned with issues such as stability, feedback, and optimality, there has been growing interest in designing algorithms that find feasible open-loop trajectories for non-linear systems. In computer graphics the problem of physically realistic simulation of mechanisms is of increasing importance.

This course will aim to cover a variety of methods for computing motion that have been developed in the research areas of discrete and heuristic search, algorithmic motion planning, optimal control and reinforcement learning. Emphasis will be given to problems that involve high-dimensional systems (e.g., manipulators), multi-agent systems (e.g., networks of moving systems), or systems with significant dynamics (e.g., flying or walking systems).

In particular, the course will be split into the following four modules:

- Module 1: Discrete and Heuristic Search
  - Review of standard search-based methods
  - Applications in navigation and manipulation challenges
  - Recomputing paths given dynamic changes
- Module 2: Algorithmic Motion Planning
  - Combinatorial solvers and potential functions
  - Sampling-based motion planners
  - Generalizations of the basic motion planning problem
- Module 3: Dealing with Dynamics - Optimal Control
  - Optimal analytical control
  - (Iterative) Linear quadratic regulator (LQR), partial feedback linearization
  - Dynamic programming foundations, value iteration and policy search
  - Stochastic optimal control
- Module 4: Reinforcement Learning
  - Stochastic gradient descent
  - Temporal difference learning
  - Function approximation
  - Policy improvement

Text and Reading Material
No textbook is required. Students are expected to take notes during the lectures and study them. Regular attendance is highly recommended. If you miss a lecture, you are responsible for all material covered or assigned in class.

The class will draw upon material that is covered by the following books:

"Principles of Robot Motion: Theory, Algorithms, and Implementations"
by H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavraki and S. Thrun MIT Press, Boston, 2005.

"Planning Algorithms"
by Steve LaValle, Cambridge University Press, 2006 Available Online.

"Dynamic Programming and Optimal Control"
by Dimitri P. Bertsekas, 3rd ed. Vols. I and II, Athena Scientific, 2007.

"Reinforcement Learning: An Introduction"
by Richard S. Sutton and Andrew G. Barto, Cambridge, MA: MIT Press, 1998.

“Heuristic Search: Theory and Applications”
by Stefan Edelkamp, Stefan Schroedl, Morgan Kaufmann; 1 edition, 2011.

Deliverable
Each student will work on an individual semester-long project corresponding to half of the course’s grade. Students are encouraged to identify topics that are related to their research efforts. Nevertheless, the project must also be related to the course material (i.e., computing motion).

In-class short quizzes reviewing material covered in class will be frequently taking place. These will be in the order of 10 to 15 minutes long in duration. They will often contain multiple choice questions or questions that can be briefly answered in a couple of lines. There are expected to be about 10 quizzes during the entire semester. The grade from the best 8 of these quizzes for each student will be used towards the final grade.

Students will also be evaluated based on the level of their participation and performance during the lectures, which primarily involves answering questions during the lecture.

Semester-Long Project
For the project, the students will be asked to work during the entire semester by preparing and frequently submitting intermediate reports and demos during the semester. A final project presentation and demo will take place during the exam week in front of the entire class. The objective of the project is to result in a final report that will be equivalent in quality to a paper ready for submission to an academic conference. The presentation should also be equivalent to a conference presentation and accompanied by a working demo. Each student will work on a separate project. The topic should be decided after coordinating with the instructor.

The students need to submit typesetted reports using LaTeX. Resources on using LaTeX are available on the course's website. You are expected to use IEEE RAS' double column format for conferences (see: `http://ras.papercept.net/conferences/support/tex.php`).

The following project components and deadlines for submissions will be used to guide your work during the entire semester:

1. September 16: Project Proposal
    - Provide a short abstract for the project
    - Then in more detail describe the general topic, provide motivation for the work and application area, emphasize the significance and the difficulty of the challenge, highlight the state-of-the-art and the main idea behind the approach you are considering, justify why your project will be novel
    - 2 to 3 pages, corresponding to the Abstract and Introduction sections of the final report
2. September 30: Literature Search
    - Improve upon the Introduction section given better knowledge of the related literature and feedback from the instructor
    - Provide a comprehensive coverage of the related literature on the topic, make sure to cover efforts from different research communities and perspectives, do not just describe related work but try to point out the similarities and differences between existing efforts, identify key contributions in the field as well as limitations of existing work, conclude with the set of open-problems given the state-of-the-art, specify which of the existing contributions your project will be building upon and which open problems it will be addressing
    - 2 to 3 pages, corresponding to the Related Work section of the final report, references are excluded from the page count
3. October 14: Experimental Setup and Infrastructure
    - Improve upon your assessment of the related literature given initial experimentation and feedback from the instructor
    - Work on providing the software/hardware platform that you will employ for your work, describe the experimental setup for your evaluation, demonstrate to the instructor the capability to run experiments or simulations on the proposed research topic using naive

solutions (even if your project is more theoretical in nature, you need to provide some form of simulation that evaluates or confirms your analysis)

- 1 to 2 pages, corresponding to the Experimental Setup section of the final report

4. October 28: Algorithmic Methodology and Mathematical Foundations

    - Improve upon your experimental platform given implementation of methods and evaluation, as well as feedback from the instructor
    - Describe in detail the novel algorithmic or mathematical process that you are proposing in order to address the project's challenge, start from the foundations of the solution in existing work and progress in emphasizing the new aspects of the approach, provide an initial implementation of the corresponding solution on the experimental platform that you have available and perform initial evaluation - no need to describe data at this point but you should be getting initial indications on how the method is performing
    - 2 to 4 pages, corresponding to the Problem Setup and Proposed Methodology sections of the final report

5. November 11: Analysis of Properties and Comparison Solutions

    - Improve upon the algorithmic solution given the observed experimental performance and analysis of the solution's properties - notice that no feedback will be provided on the previous report and you need to show the capability of improving upon your proposed solution given additional study and experimentation
    - Evaluate the properties of the proposed methodology (e.g., in terms of metrics such as safety, completeness, optimality, computational performance and complexity, scalability, information requirements, etc.), theoretical and mathematically study of the underlying properties is highly encouraged if it makes sense for the project, describe alternative methodologies that need to be evaluated in order to show the relative performance of the proposed approach, provide initial implementations of the alternative methodologies and initial evaluations
    - 2 to 4 pages, corresponding to the Properties and/or Comparison Solutions sections of the final report - note that more theoretical projects will be focusing more on "properties", while more experimental ones will be focusing more on evaluating performance against "comparison solutions"

6. November 25: Experimental Evaluation

    - Improve upon the analysis of the proposed method, as well as better identify appropriate comparison points given additional study, experimentation and feedback from the instructor
    - Perform a comprehensive experimental evaluation of the proposed methodology in appropriate benchmarks that you have identified and described in detail, the evaluation will include running experiments with competitive methods and providing the corresponding graphs that show the differences between the performance of different algorithms, study experimentally the performance of the studied methods in terms of the appropriate metrics (e.g., safety, completeness, optimality, computational performance and complexity, scalability, information requirements, etc.)
    - 2 to 5 pages, corresponding to the Results section of the final report, even theoretical projects need to evaluate methods using some form of simulation

7. December 9: Discussion of Performance, Improvement of Methodology and Revisions

    - Improve upon the experimental evaluation given a study of the resulting graphs and feedback from the instructor

- Provide a detailed discussion of the observations during the experimental evaluation, based on your conclusions, revise and improve the proposed methodology, run additional experiments and show potential improvements against alternatives or versus prior performance, study in terms of theoretical properties the effects of your revisions in the underlying algorithm, update the introductory part of the paper given observations and the potentially shifted focus of the project, prepare the reader of the report about the actual contribution of the project and identify the important future steps
- 2 to 3 pages, corresponding to the Discussion section of the final report, as well as revisions and extensions in the entire paper given observations, further study and changes in the underlying methods - at this point you should have an initial version of the complete report ranging between 13 to 24 pages excluding references.

8. December 16: Final Report and Presentation
    - Improve upon the entire paper and tighten up description so as to bring the size of your final report down to 12 pages including references. No feedback will be provided on the previous longer version of your final report (but it will be graded together with the final report). The final report will no longer need to describe in detail the process with which you ended up in the best performing solution (i.e., the initial approach you were considering and how you ended up improving upon it). It needs to directly inform the reader about the final contribution and observations that will be the result of your entire effort. It still important, however, to provide key comparisons with alternatives in your experimental setup and justify the algorithmic choices made in the final solution.

Submissions will be processed through Sakai. Late project report will not be accepted. Final project presentations cannot be easily rescheduled. Unless there are unprecedented circumstances, no incomplete grade will be awarded and students will be graded based on the submitted reports and presentations up to the exam week.

Extra credit can be awarded for impressive effort and performance in the project. Such projects would correspond to reports and demos that can be easily submitted to an academic conference.

Grading System
The final grade will be computed according to the following rule **(this is tentative)**:

Project Proposal (relevance, significance, interest, novelty and level of difficulty of proposed project)	5 points
Literature Search (comprehensiveness, capability to classify previous work, ability to identify relations and open problems)	5 points
Experimental Setup (appropriateness of platform, benchmarks and capability to evaluate the correct metrics for the challenge and the methodologies)	5 points
Algorithmic Methodology (formal and rigorous description, appropriateness of solution, clarity of presentation, novelty of ideas)	5 points
Analysis of Properties (proper identification of comparison points, mathematical capabilities in analyzing proposed methods)	5 points
Experimental Evaluation (comprehensiveness of evaluation, study of appropriate metrics and informative presentation of data)	5 points
Discussion of Performance (capability to adapt to observations and appropriately revise solutions, proper identification of future research steps)	5 points
Final Report (ability to communicate message and progress the state-of-the-art, novelty, significance, difficulty, success in objectives)	7.5 points
Final Presentation/Demo (clear explanation of objectives, methods and results, ability to maintain interest, use of visual assistance, success of demonstration - feedback from all the students will be used to grade the presentation)	7.5 points
In-class quizzes (best 8 out of 10):	40 points
In-class participation:	10 points

As a **rough** guide, the following rule may be used for the final grade **(this is tentative)**:

Final Grade	A	B+	B	C+	C	D	F
Range	$\geq$ 86	77-85	68-76	59-67	50-58	41-49	$\leq$ 40

Academic Standards

Discussions on the class material, on project challenges and programming practices are highly encouraged. Furthermore, students should try to exchange feedback on drafts of their project reports. Nevertheless, each student needs to eventually independently develop one's proposed methodology, code the corresponding solution and execute the corresponding experiments.

Quizzes are strictly individual efforts and no collaboration is allowed.

You should carefully study the website of Rutgers University on Academic Integrity and the corresponding policy, as well as the corresponding website from the department of Computer Science. Links are available through the course website. Your enrollment in this course implies that you have read these policies, and that you subscribe to the principles stated therein.