[EECS C106A] Fall 20 Potential Project Ideas

Compiled by your friendly neighborhood (U)GSIs: Amay Saxena, Tiffany Cappellari, Jaeyun Stella Seo, Isabella Huang, Ron Thalanki, Donggun Lee, Jay Monga, Kyoungtae Lee, and Ritika Shrivastava

1. **General Project Overview**

Final Projects will be completed in groups of approximately 4 students. If you wish to work in a much larger or smaller group, you will be expected to explain why in your Project Mini-proposal. This year’s final projects will look quite different from semesters before. This semester, we are introducing 3 tracks for final projects: simulation projects, hardware projects, and design sprint projects. All groups will have a supplementary budget (to be determined) for any costs associated with the project. Some sample project ideas have been outlined below.

1. **Simulation Projects**

Simulation Projects will be projects built in simulation environments, like Gazebo. Because you’re not limited by hardware, you can think up some truly crazy ideas to try! Please note that as you consider these projects, you should consider what simulation environments you will need to explore as well as what you might need to build.

Sample Projects:

1. **Sound Based Search/Ping Pong Ball Collector**

Ping pong balls make a pretty distinct noise when bouncing and it’s annoying to pick them up normally. Use audio-based sensing to locate and collect one or multiple bouncing ping pong balls in a 2D space. For our purposes, a turtlebot should be sufficient. The environment could contain ambient noise and noise from other sources like people talking or walking around. Other types of sensing may be used if needed, but sound sensing must remain an essential part of the system (removal of sound-based sensing would cause the system to fail). I see a lot of room for experimentation in terms of multiple microphones in the environment, only microphones on the robot, level of ambient noise, multiple objects falling simultaneously, level of light/effectiveness of cameras, obstacles and environment, etc. A big issue with a real-world implementation of this is that people may accidentally step on the thing, so incorporating some kind of solution that involves avoiding people would be pretty nice to see.

This project would likely be done in the STDR Simulator. You would need to figure out how to make point sources/sinks and maybe do your own 2D audio simulation. Detecting each bounce may require some amount of fourier analysis or maybe something more/less advanced, but figuring out how to deal with multiple bouncing balls will be an interesting challenge.

1. **Frogger/Jay-walking Robot**

You will be implementing the arcade game frogger in a simulation environment (Gazebo or STDR) where a robot will need a cross street avoiding obstacles that will most likely be cars, but use your imagination. The robot should have to use its own sensing to perceive obstacles and avoid them, and be autonomously controlled. A turtlebot should suffice, but once again, feel free to use your imagination. In simulation, the opportunities are endless.In addition, the robot should be slow enough in comparison to the cars to make the game challenging, but not impossible. There will be a lot of room for experimentation in terms of difficulty in perceiving the obstacles/cars and strategies to avoid them, and potential to utilize some game theory.

If you’re up to it, the cars can all be simulated and place an AR tag on a physical robot or yourself to play the game. In this approach, manual perception, planning, and control is fine as the main task would be to just get the simulated and real world environments interfacing well with each other. Just don’t play in the street.

1. **Simulated Drawing Robot**

A staple of 106A projects has always been the artist robot; a Baxter or other bot equipped with a drawing utensil capable of making artistic masterpieces. You will be continuing on the legacy by designing your own drawing simulation within Gazebo. The approach that comes to my mind would be to measure orientation of a utensil and the force it applies to some surface, and then translate that into an actual image, but use your imagination! After this system is set up, use a Baxter or another robot with an arm-like manipulator to actually make some drawings using the system you designed. The proper grasping and control of some kind of drawing utensil will probably be the second significant challenge after the drawing simulation; be ready to take a sneak peak of concepts like “force closure” which will be explored more in 106B. I wanted to have the robot take in a latex document and manually render it, but it would be acceptable to have the robot draw out any given image picture input.

1. **Modeling the Human Body (joint with METRiCS Lab, UCSF)**

Modeling the human body and the relationship between joints and limbs and how they move--it’s all a very complicated process. In this project, you’ll work with Rob, a former C106A GSI, to try to build a model of the body using data from consumer electronic sensors (think: Xbox Kinect, iPads). You might be able to cross-validate your model with state of the art motion capture systems as well.

UPDATE: Rob has been very kind and has offered a few different ideas on what interesting work can be done in this space. Check out his prompts [here](https://drive.google.com/file/d/1lqf52qKS0BSj043A8EsnUfPYmSK6t7LM/view?usp=sharing).

1. **ROAR**

ROAR stands for Robot Open Autonomous Racing and is Berkeley’s first AI race car competition. With most of the semester being online, students can use Carla (an autonomous driving software) to race on tracks in all environments (icy & sandy). Use Carla to simulate various racing environments and train a controller that can accomplish several tasks. These tasks include: following traffic signals, Multicar interaction, real world mapping with Realsense / LIDAR and autonomous driving with Autoware. This project also can have a physical aspect if you are located in Berkeley. Please email Ritika for more information. ritishri at berkeley.

1. **ROAR + Focus on Racing**

Use Carla to train a simulated car to go autonomously as fast as it can on a track, in the context of the following problem: the car is allowed to take one lap of the track slowly and collect information about the track (such as curvature, friction coefficient, obstacles, turns etc) and then a second lab where the car utilizes the information it learned from the test lap to go as fast as possible. This project is similar to the ROAR project above and revolves around inferring features of the track from sensors and your own motion to facilitate track memorization to allow for drifting and racing. The project focuses on making your simulated car go as fast as possible on the race track.

1. **Table Tennis Ball Bouncer / Tennis Ball bouncer**

Bouncing a ball on a paddle / racket is the first task that any new player learns. Let’s make a robot that can bounce a table tennis ball on a paddle for the longest time! For this project, you will be writing an algorithm to track the position of the table tennis ball and map the trajectory of the ball. Based off this trajectory your team will have to move the Baxter/Sawyer’s arm to the position where its paddle can bounce the ball. Goal is to make the ball bounce as many times as possible!

<https://www.youtube.com/watch?v=flyEY7P48D0&feature=share>

1. **Robotic Arm pushing a Box on Various Surfaces**

Pushing a box on a table is easy with a robot. Let’s make this a bit more complicated by adding two other tables of different frictions. The robot has to push the box / item past a point without pushing it over the edge of the table. It also has to make sure that the box reaches the point. The robotic arm can have a RGB camera to assess where the object is on the table. The final goal is to get the fastest time it takes for a robot to assess the friction of the table and push the box to that point without pushing it over.

<https://www.youtube.com/watch?v=OKOyTQQcrLw&feature=share>

1. **Semi-autonomous surgery**

Da Vinci Surgical System is a teleoperation system in which a surgeon uses a master device to control slave systems that operates surgery on patients. In this project, We automate a part of surgery, such as cutting tumors, grasping organs and so on.

<https://www.youtube.com/watch?v=GWzrwiG9cS4&ab_channel=daVinciSurgery>

In this automation, the motion of the slaves must satisfy the safety conditions, such as not cutting other organs. We could get some help from Prof. Ken Goldberg.

1. **Window cleaning robot for skyscrapers**

In this project, we design a cleaning robot for skyscrapers. Currently, people are employed to clean the windows.

<https://www.youtube.com/watch?v=wcRuXrduj6Q&ab_channel=BBC>

In this project, we 1) conceptually design a robot that equips all tools for cleaning and 2) find a path that covers all surfaces of skyscrapers.

Bonus: Open windows, weird curvature

[2D simulation, with effect of gravity in one direction acting on all robots]

1. **Drone light show!**

Have >10 drones in Gazebo and manipulate them to make an alphabet in space. When the user inputs the alphabet, the locations of each drone are calculated and transmitted to them. Each drone has to maintain the position well even with the wind coming from other drones or constant wind in the environment. It would be more challenging if it is done *without any central system* that calculates and assigns the position to each drone.

1. **Robot riding a swing**

Riding a swing is a very sophisticated job, as the rider has to manipulate various muscles organically. Students will simulate a humanoid robot and the goal is to ride a swing well. Swinging more than a certain angle would be a plus point. Students can use a very simple machine learning (reinforcement) to control each joint by setting the objective function to be the angle. Students can come up with algorithms that controls each joint to maximize the angle of the swing.

1. **Piano Movers’ Problem**

A student group can use a simulation of turtlebots in order to manipulate a larger object in 2 dimensions (planar manipulation). A research paper that focused on having robots rotate a couch can be found [here](https://www.researchgate.net/profile/Daniela_Rus/publication/2605800_Moving_Furniture_with_Teams_of_Autonomous_Robots/links/56d0437e08ae059e375cfaa8/Moving-Furniture-with-Teams-of-Autonomous-Robots.pdf) for reference. Students can simulate a large box for simplicity and have 2-3 turtlebots navigate the object around obstacles to reach some goal.

1. **Object manipulation in 3D with millirobots**

Tiffany’s research project’s goal is to have a small team (3-4) of millirobots cooperate in order to move a rigid object in a 3 dimensional space. For this project, the object will be a fallen chair and the robots’ goal is to upright it. This project will be using V-REP and ROS as well as simulations of Kamigami robots. As the project is still in its very early stages (and subject to change) the exact project guidelines for a 106A/206A project group is still yet to be determined.

1. **Reactive Control Improvisation Algorithm Simulation**

Try implementing Daniel Fremont’s Reactive Control Improvisation algorithm <https://arxiv.org/abs/1804.05037>  
  
The algorithm is implemented in a game-like format in a discrete space. An improviser and an adversary will take turns moving across a “game board”. The improviser’s goal is to visit a set of waypoints exactly once per waypoint while avoiding the adversary. The adversary will also be moving across the board and the improviser will need to react to the adversary’s movements and improvise its next move accordingly. For an extra challenge, you can have the adversary have a goal of “catching” the improviser.

1. **Multi-robot competition in the absence of central communication**

Have two teams of mobile robots compete with each other in a maze-like (or otherwise obstacle-rich) environment where there is no centralized communication between all robots. So every robot does not know where every other robot is at all times. You can have them play a game like capture the flag or devise a game of your own.

You may experiment with different communication paradigms:

* Robots never know each other's location until they are within sensing range.
* Robots on the same team can communicate with each other and share their location but cannot see the other team until they are within sensing range.
* All robots always know where other robots are.

You may experiment with different environment-knowledge paradigms:

* The robots all have a perfect map of the environment and have perfect knowledge of their own location.
* The robots all have a perfect map of the environment, but do not have perfect knowledge of their own location, and must use sensors to localize themselves.
* The robots do not have any prior knowledge of their environment.

Every mix of the above paradigms will require different strategies on the parts of the players. You could use STDR simulator (from Lab 4) as your simulation environment as it is a light-weight way of simulating multiple mobile robots with laser scan sensors in custom environments. You could of course, also use Gazebo. You could also get fancier and do a similar project where your agents are drones instead of grounded mobile robots which would add an additional layer of complexity. You can also experiment with having a team of robots compete with a human player who is controlling the other team.

1. **Multi-robot collaboration in the absence of central communication**

Have multiple mobile robots collaborate on a task such as an explore-and-rescue operation in a complex environment. Experiment with various communication and environment-knowledge paradigms. The problem of multi-robot exploration is an interesting one, because with multiple robots you can split up the task of exploring a large space, but this comes with a whole host of algorithmic challenges. For instance, how do you assign new spaces for each robot to explore in a way that maximizes some measure of efficiency? How do you merge the information collected by each robot into a single, global map? The problem can be further compounded when the robots have no means of central communication, as is the case with real life search-and-rescue scenarios where establishing a central network is difficult. Once again, there are many methods you can choose to simulate these scenarios: STDR simulator for 2D multi-robot simulations with laser scans, or more complex scenarios with drones in Gazebo (or in a custom simulation of your own!).

1. **Human-robot interaction in augmented reality**

Have humans collaborate (or compete) with a robot in an augmented reality setting where both the task and the robot exist in AR, and the human interacts with them through a smartphone. You can use off the shelf AR libraries like ARCore for Android or ARKit for iOS to get information about the environment and to place your simulated robot in AR. You can then have the simulated robot and the person perform tasks such as navigating the simulated robot around real obstacles in the person's environment to get to some goal. You can also look at [OpenARK](https://ptolemy.berkeley.edu/projects/augcog/OpenARK.html) which is an open source AR library developed right here at Berkeley in the Augmented Cognition lab. OpenARK provides a bunch of advanced functionality such as multi-object tracking, hand detection, plane detection, and full visual-inertial SLAM.

1. **Robot Soccer**

Simulate a game of soccer between two autonomous robot teams in a 2D simulation environment of your choice. Things to consider are:

* + Ball sensing paradigm: how do players collect information about the location of the ball? Do they all always have perfect knowledge, or do they only know where the ball is when they point a laser scanner toward it, or sense it in some other way?
  + Communication paradigm: how much information do teammates share? Do they all always share their current locations and strategies with each other or must the robot sense each other's location?
  + Path planning and collaboration: how do the robots navigate to a moving ball and intercept in such a way as to deflect it to a desired direction? How do teammates collaborate with each other to play defensively or offensively to score goals?

1. **Manipulation of soft bodied objects in simulation**

Using a physics simulator with soft body support (eg Pybullet), perform simple manipulation of some soft objects. Stick to simple objects like rope and cloth. Use some basic planning to perform a sequence of action primitives (eg. fold a cloth in half a couple of times, roughly coil a rope, create a loop with a rope, pull the rope’s end through a loop to create a knot (!)). Can assume full state observability depending on complexity of final task.

1. **Robot rendezvous with local mapping**

Inspired by real world challenges in search robotics, where multiple search and rescue robots may be deployed from different starting positions. Often, these robots will need to rendezvous (this can be necessary when multiple robots need to be in the same place to provide assistance to a victim, or perform decluttering, etc). Build a test environment in a simulator like Gazebo, and have multiple robots explore the area from different initial starting points. Build a centralized “brain” that synthesizes all the local maps gathered by each robot (by way of SLAM). Would be nice to not assume the existence of a reliable positioning system. Then, when one robot signals that it wants to initiate a rendezvous, demonstrate that the other agents can use this synthesized information and successfully make their way to the meeting point.

1. **Goal inference for an adversary agent**

Build a simple game in simulation, where you control a character/agent and try to tag any one of N discrete goal points in a virtual environment in order to win the game. However, build an adversary agent that is not only a bit faster than you, but performs goal inference to try to intercept you at whichever goal point you’re trying to reach. If the agent hits the goal point first, it disappears. The agent should be pretty smart and adapt to ways in which you’d try to outsmart it. There are papers on goal inference and nice ways to implement it that will work well here.

<https://www.ri.cmu.edu/pub_files/2013/3/legiilitypredictabilityIEEE.pdf> Goal inference algorithm in section IVA. Goal inference is based on knowing that optimal trajectories towards a goal is governed by a cost function that is known. A simple cost function would be distance traveled -- here for example, the best approach towards a goal point is to take the most direct path.

1. **3D head pose estimation from 2D webcam stream**

Inspired by applications in robotic assistive feeding. Using simple CV techniques, are you able to construct (just from a webcam stream of your face) an estimate of your head + mouth pose in 3D? Would be great to see a real time visualization of your 3D pose estimate in a virtual environment. Bonus points for implementing the following: given a 3D head + mouth pose, move a robot arm in that same virtual environment in a way that it directs food directly into your mouth. Can use AR tags stuck to your face as a ground truth.

1. **Ping pong ball-hitting robot (Wii Sports but AR Tags)**

Since this seems to be suggested almost every year but neither safe nor feasible in real life, could be fun in simulation. Shoot balls at random onto a ping pong table setting, and have a robot return your shot. Should be interesting because joint position and velocity limits concerns are thrown out the window. As a first pass, assume complete state observability. Stretch goal is to work with only partial state observability, as when the robot is only equipped with an RGBD camera to track and hit the ball. Another stretch goal- to serve the first shot to the robot, stick an AR tag to a paddle in real life, perform a hitting motion, and initialize the ping pong ball with the starting velocity as if it was hit in real life by that paddle motion.

1. **Indoor navigation**

Students will make an indoor navigating robot in gazebo using a camera & IR sensor. The goal is to make a map of the indoor space, like an office or house. The robot starts to roam and collect data like the current position and distance from the wall. It spontaneously moves to undiscovered locations. The robot has to stop navigating and report when it thinks it has explored everywhere.

1. **Virtual Reality and Haptics (HART Lab)**

Work with a novel haptic feedback surface and IMU + depth cloud data and build software & VR interfaces to enable experiments on tactile feedback in virtual reality. Use classical computer vision techniques to downsample image data to display on low-resolution haptic surface.

1. **Grasping-Related Projects**

With grasp quality metrics, can you create grasp approach trajectories and contact point selections for any given object?

1. **NEW: Path Planning with Valmik**

In the second half of the course, you'll learn about path planning algorithms for robot manipulators, including the Rapidly-exploring Random Tree algorithm (RRT). This algorithm is ubiquitous in industry, but applying it to systems with nonholonomic constraints, such as turtlebots, cars, and other mobile robots is still an open research problem. I'm working with Prof Sastry to develop an RRT-based path planning algorithm for mobile robots using some modern nonlinear control techniques, and I would like some 106A students to implement it in a high-quality simulator such as Unity. Your group would re-implement the algorithm in the language of your choice, design a visualizer in Unity or another simulation/graphics engine, and test the system on a number of tasks such as maze-solving and parallel parking. If you're interested, you could also experiment with some improvements to the algorithm, such as path-biasing, alternate cost functions, etc.

The ideal group would have some experience developing aesthetic things in Unity or a similar environment, be relatively comfortable with the math in 106A, and be interested in learning more about path planning. No nonlinear controls knowledge required.

1. **Hardware Projects**

Hardware Projects will be projects that have a strong mechanical component involved. We’d love folks to be able to work with hardware if they want, so we’ve outlined a few options below.

Sample Projects:

1. **MicroServo Robot**

Build your own robot that can mimic the actions done to its model component. Has a teaching and learning component. There are a lot of tutorials and can be done with a cheap set of materials.

Use the Microservo robot you create to grasp an interesting project or add an additional sensing / control aspect to make your project unique.

* <https://www.youtube.com/watch?time_continue=88&v=bLnAJ-mSElE&feature=emb_logo>

1. **Table Tennis Ball Bouncer / Tennis Ball bouncer**

Bouncing a ball on a paddle / racket is the first task that any new player learns. Let’s make a robot that can bounce a table tennis ball on a paddle for the longest time! For this project, you will be creating a robot that can bounce a ball. You will also write an algorithm to track the position of the table tennis ball and map the trajectory of the ball. Goal is to make the ball bounce as many times as possible!

* <https://www.youtube.com/watch?v=lYyAMDYzJQM&feature=share>
* <https://www.youtube.com/watch?v=110QByzrSAY&feature=share>

1. **Arduino Controlled Hydraulic Powered Robot**

The idea came from <https://www.youtube.com/watch?v=P2r9U4wkjcc>. The great thing about this video is that most tools seen in the video can be found around the household or acquired at a low price. The best way to expand on this project is to control the robot using an arduino. With this robot that you can now control, make the Hydraulic robot pick an object and place it in the desired location.

1. **Design of light-weight haptic device**

Haptic devices provide force feedback generated in virtual reality. It is important to design a light-weight haptic device as much as possible. In this project, we investigate a few haptic devices and design a new light-weight device.

[Haptic Devices](http://www.hitl.washington.edu/people/tfurness/courses/inde543/READINGS-03/BERKLEY/White%20Paper%20-%20Haptic%20Devices.pdf)

1. **Kamigami (& other toy things) + arduino/ESP32/Raspberry Pi**

Kamigami robots are cheap toy robots available for purchase on Amazon and can be used for a low budget hardware based project along with an arduino, ESP32, or a Raspberry Pi and a camera or sensor (if using a Raspberry Pi, we recommend a Raspberry Pi 0 and a Pi camera). Some ideas include:

* Having the robot find, identify, and obtain a user-specified object.
* Having multiple robots perform a “flash mob”-esque coordinated performance/task.
* Having the robot(s) perform some sort of “search and rescue” task.

1. **ROAR + AutoX**  
   Do you have the need--the need for speed? Stella certainly does. She misses [autocross events](https://www.scca.com/pages/what-is-autocross), where drivers will try to finish a course as fast as they possibly can. Can you help Stella relive her autocross joys from before she sold her sporty car for a more reasonable sedan? But also on a smaller scale? For this project, consider building a controller for the ROAR cars. You can rig up a steering wheel and some pedals to communicate with one of the ROAR cars so that you can tele-operate a ROAR car as if you were really driving it. This likely will end up becoming mostly a controls project.
2. **Design Sprint Projects**

Design Sprint Projects will be different from Simulation and Hardware Projects. In this style of project, you’ll identify a problem, engage with the community who faces this problem, prototype a solution, iterate on this solution, and then provide a report on what your suggested next steps are. Do not be fooled into thinking that this might be an easier project, especially in terms of time commitment! Good design requires rigorous testing and community engagement.

Sample Projects:

1. **Permobil** (Pending details from Permobil)   
   [Permobil](https://permobilus.com/about/) is a company that takes a personalized approach to wheelchairs and has made great strides in comfort and innovation. You’ll work with an engineer from Permobil to identify needs in the community they serve and prototype a solution. You’ll continue to engage with the community to test your prototype and iterate upon your idea.
2. **Docking / context awareness of, and response to, environment such as tables, doorways, and ramps.**

At present, wheelchairs have little or no awareness of the surrounding environment and rely entirely on the user for guidance. However, not all users perform fine-motor manipulation and may lack the head mobility to look around the chair to observe the immediate surroundings. This may lead to personal injury and property/chair damage due to collisions and rollovers. This project would add a cooperative degree of autonomy to the wheelchair to aid with tasks such as positioning with respect to a table and guidance through doorways and along ramps. Initial development would likely be done in a virtual environment with a wheelchair model, though an ambitious team can have access to one of Permobil’s power wheelchairs if desired.

1. **Motion planning of 7 DoF Arm + PWC base**

Some wheelchairs are fitted with a robotic arm to aid with user activities of daily living (ADLs). In most configurations the arm and wheelchair act as independent units. This project would explore a combination motion planning controller that utilizes the capability of having a robotic arm mounted on a mobile base. The path planning of the arm needs to have simple and intuitive inputs that can be adapted to, and the operational space and path planning should have restricted zones to avoid user and other hardware collisions.

1. **Dynamics planning and AI-based adjustment of power wheelchair driving behavior.**

Due to caster dynamics plus variability of user position / changing center of gravity location based on environment / slope and position of actuators, driving a wheelchair can become a complex task. This project would explore modeling of the wheelchair dynamics so that they can be effectively compensated to provide a smoother and more intuitive user experience. Compensation techniques can explore machine learning and artificial intelligence decision making.

1. **Physical Therapy and Rehab (UCSF: SF, Oakland campuses)**There is a lot of room for innovation in the field of medicine, which is partially why biomedical engineering is such a hot topic these days. But you can make a difference even without an extensive background in biology! Lots of clinicians need help with making their care engaging so patients will participate. You can imagine that this is particularly true of those working in Physical Therapy (PT) and Rehab. For this project, you’ll look into how you can make a difference in PT and Sports Medicine with folks at UCSF.
2. **Point of Care Projects**  
   The age of tele-medicine has been brought upon us sooner than expected with the novel coronavirus. One of the key aspects of tele-medicine is being able to get the physiological measurements that you need without heading into a doctors’ office. In this project, identify a field of healthcare that could use some innovation, and provide a prototype of a point-of-care device. This could be a makeshift EKG, a pulse-oximeter, sphygmomanometer. Or maybe you can improve upon spirometry (important since the novel coronavirus is a largely respiratory disease) or diagnostics for heart murmurs. Talk with Stella and Professor Sastry, and we’ll try to get you connected with clinicians in the field of your interest.   
     
   Note: BioEs may find this an interesting stepping-off point for BioE 101 or 192.
3. **Underwater exploration using remotely operated underwater vehicle (ROV) (under contact with Oceaneering, https://www.oceaneering.com/rov-services/)**

ROVs explore underwater environments in 3D and perform tasks such as assembly.

<https://www.youtube.com/watch?v=83UsvUFVflg&ab_channel=iXblue>

In this project, we simulate the underwater environment and perform a task such as assembly (connecting two pipes). Using gazebo, we simulate underwater dynamics and robots doing tasks. You can look into what underwater tasks need doing and how we can safely do them autonomously!

<https://www.youtube.com/watch?v=Va9sO7S03K8&ab_channel=SimaoSilva>