*AA 598B Special Topics*

# Decision-Making & Control for Safe Interactive Autonomy

Instructor: Prof. Karen Leung

*Autumn 2024*

https://faculty.washington.edu/kymleung/aa598/

# Announcements

- Link to recording on website
- Long paper discussion on Wednesday
  - Submit short paper summaries

| | |
|---|---|
| [Paper 3] Fredovich-Keil, Bajscy et al 2020 Confidence-aware prediction & planning | Tyler Han |
| | Tommy Zhang |
| | |
| [Paper 4] Burger et al 2022 Interaction-Aware Game-Theoretic Motion Planning | Sayali Nehul |
| | Richard Feng |
| | Jeffrey Justin |

- Project proposal due Nov 1 Friday
  - More instructions posted on website
- Guest lecture by Prof. David Fridovich-Keil from UT Austin next Monday
- Moving OH to Monday 12PM – 1PM

# Last time

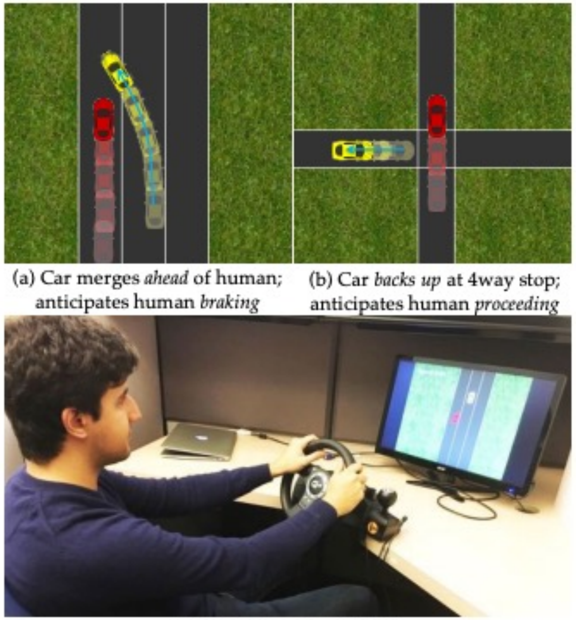
- Introduced general trajectory planning problem
  - Problem with robot only
  - Problem with robot + humans
    - Objective becomes a random variable
    - Constraints become a random variable
    - Possible outcomes can be very distinct due to multimodality

# Today

- Discuss various planning approaches
  - Socially-aware approaches
  - Solution methods
- Brief introduction to game theory
  - Overview basic concepts in preparation for guest lecture next week

# *Socially-aware* planning

# Planning with access to other agent's reward



(a) Car merges *ahead* of human; anticipates human *braking*

(b) Car *backs up* at 4way stop; anticipates human *proceeding*

(c) User drives human car

**Fig. 1:** We enable cars to plan with a model of how human drivers would react to the car's actions. We test the planner in a user study, where the car figures out that (a) it can merge in front of a human and that will slow them down, or (b) it can back up slightly at an intersection and that will make the human go first.

At every iteration, the robot needs to find the $\mathbf{u}_{\mathcal{R}}$ that maximizes this reward:

$$\mathbf{u}_{\mathcal{R}}^* = \arg\max_{\mathbf{u}_{\mathcal{R}}} R_{\mathcal{R}}(x^0, \mathbf{u}_{\mathcal{R}}, \mathbf{u}_{\mathcal{H}}^*(x^0, \mathbf{u}_{\mathcal{R}})) \quad (5)$$

Here, $\mathbf{u}_{\mathcal{H}}^*(x^0, \mathbf{u}_{\mathcal{R}})$ is what the human would do over the next $N$ steps if the robot were to execute $\mathbf{u}_{\mathcal{R}}$.

The robot does not actually know $\mathbf{u}_{\mathcal{H}}^*$, but in the next section we propose a *model* for the human behavior that the robot can use, along with an approximation to make (5) tractable.

$$\mathbf{u}_{\mathcal{H}}^*(x^0, \mathbf{u}_{\mathcal{R}}) = \arg\max_{\mathbf{u}_{\mathcal{H}}} \underline{R_{\mathcal{H}}}(x^0, \mathbf{u}_{\mathcal{R}}, \mathbf{u}_{\mathcal{H}})$$

human reward
learned via MaxEnt-IRL

Sadigh et al 2016

# Planning with access to other agent's reward

To apply L-BFGS, we need the gradient of (5) with respect to $\mathbf{u}_{\mathcal{R}}$:

$$\frac{\partial R_{\mathcal{R}}}{\partial \mathbf{u}_{\mathcal{R}}} = \frac{\partial R_{\mathcal{R}}}{\partial \mathbf{u}_{\mathcal{H}}} \frac{\partial \mathbf{u}^*_{\mathcal{H}}}{\partial \mathbf{u}_{\mathcal{R}}} + \frac{\partial R_{\mathcal{R}}}{\partial \mathbf{u}_{\mathcal{R}}} \tag{11}$$

$\frac{\partial R_{\mathcal{R}}}{\partial \mathbf{u}_{\mathcal{H}}}$ and $\frac{\partial R_{\mathcal{R}}}{\partial \mathbf{u}_{\mathcal{R}}}$ can both be computed symbolically through backward propogation, as we have a representation of $R_{\mathcal{R}}$ in terms of $\mathbf{u}_{\mathcal{H}}$ and $\mathbf{u}_{\mathcal{R}}$. For $\frac{\partial \mathbf{u}^*_{\mathcal{H}}}{\partial \mathbf{u}_{\mathcal{R}}}$, we use that $\mathbf{u}^*_{\mathcal{H}}$ is the minimum from (10), which means that the gradient of $R_{\mathcal{H}}$ evaluated at $\mathbf{u}^*_{\mathcal{H}}$ is 0:

$u_H^*$ is optimal $\Rightarrow \frac{\partial R_H}{\partial u_H} = 0$

$$\frac{\partial R_{\mathcal{H}}}{\partial \mathbf{u}_{\mathcal{H}}}(x^0, \mathbf{u}_{\mathcal{R}}, \mathbf{u}^*_{\mathcal{H}}(x^0, \mathbf{u}_{\mathcal{R}})) = 0 \tag{12}$$

Now, we can differentiate the expression in equation (12) with respect to $\mathbf{u}_{\mathcal{R}}$:

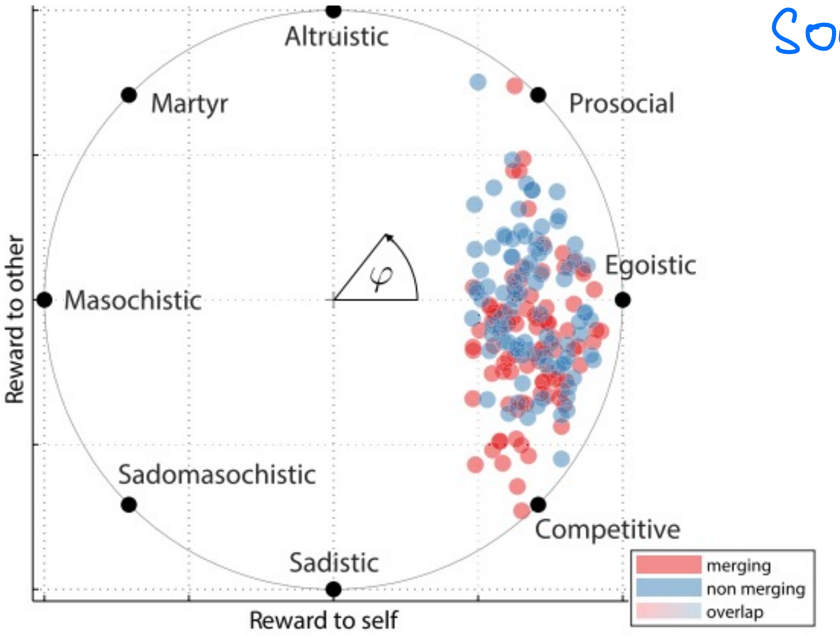
$$\frac{\partial^2 R_{\mathcal{H}}}{\partial \mathbf{u}^2_{\mathcal{H}}} \frac{\partial \mathbf{u}^*_{\mathcal{H}}}{\partial \mathbf{u}_{\mathcal{R}}} + \frac{\partial^2 R_{\mathcal{H}}}{\partial \mathbf{u}_{\mathcal{H}} \partial \mathbf{u}_{\mathcal{R}}} \frac{\partial \mathbf{u}_{\mathcal{R}}}{\partial \mathbf{u}_{\mathcal{R}}} = 0 \tag{13}$$

Finally, we can solve for a symbolic expression for $\frac{\partial \mathbf{u}^*_{\mathcal{H}}}{\partial \mathbf{u}_{\mathcal{R}}}$:

# Planning with social considerations

## *Social behavior for autonomous vehicles*

Social Value Orientation (SVO)

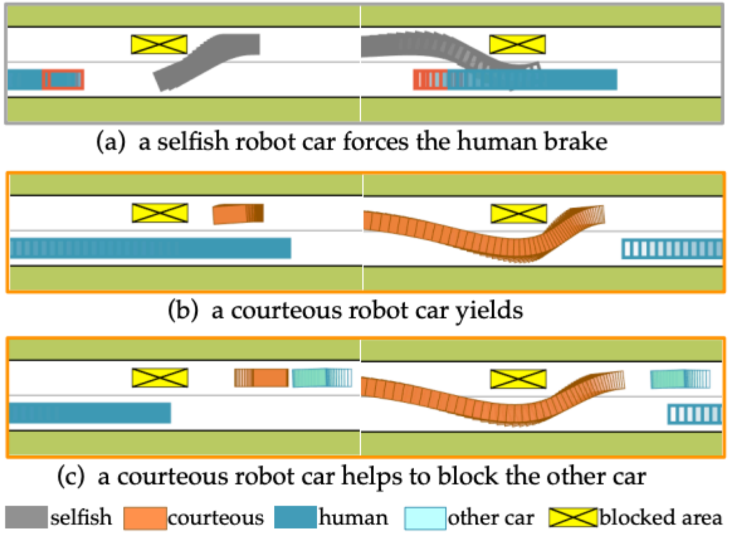



$$g_1 = \cos(\varphi_1)\overset{\text{robot}}{r_1(\cdot)} + \sin(\varphi_1)\overset{\text{human}}{r_2(\cdot)},$$

Schwarting et al 2019

# Planning with social considerations

## *Courteous Autonomous Cars*




(a) a selfish robot car forces the human brake

(b) a courteous robot car yields

(c) a courteous robot car helps to block the other car

$$C_{\mathcal{R}}\left(x^t, \mathbf{u}_{\mathcal{R}}, \mathbf{u}_{\mathcal{H}}; \theta_{\mathcal{R}}, \theta_{\mathcal{H}}, \lambda_c\right) = C_{\mathcal{R}}^{self}\left(x^t, \mathbf{u}_{\mathcal{R}}, \mathbf{u}_{\mathcal{H}}; \theta_{\mathcal{R}}\right) + \lambda_c \; C_{\mathcal{R}}^{court}\left(x^t, \mathbf{u}_{\mathcal{R}}, \mathbf{u}_{\mathcal{H}}; \theta_{\mathcal{H}}\right)$$

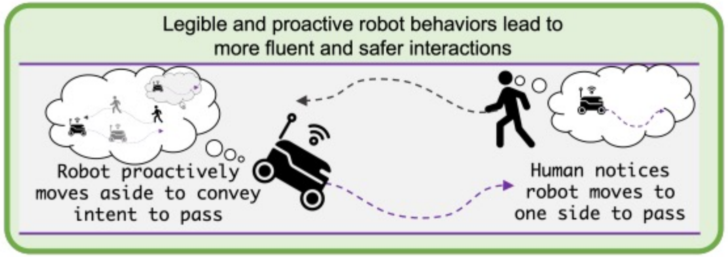
Sun et al 2018

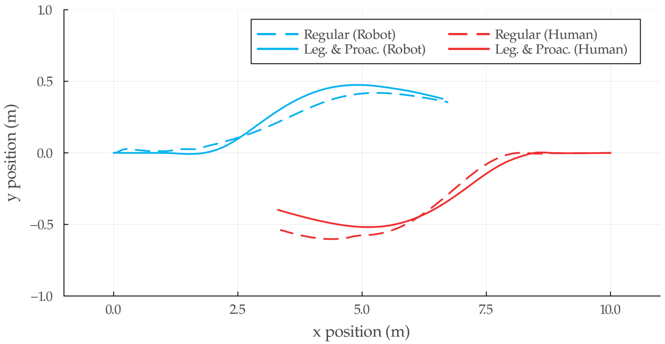
courtesy definition

1. come up with an alternate future
   eg. - robot maintains speed
   - robot "disappears"
2. for that future - what the human would do
   $J_{hum}^{alt}$
3. $C_R^{court} = J_{hum}(\ldots u_R \ldots) - J_{hum}^{alt}$

# Planning with social considerations

*Legible and Proactive Robot Planning for Prosocial Human-Robot Interactions*




$$J_{\text{incon}}(\mathbf{x}_F^{0:T+1}, \mathbf{u}_F^{0:T}) \leq \beta_F$$

Geldenbott et al 2024

cost:

$$J(\vec{x},\vec{u}) = \sum_{t=0}^{T-1} \mu^t J_t(x_t,u_t) + J_T(x_T)$$

$\mu > 1$ , $\mu$: markup

# Sequential Convex Programming

(Homework 2 problem)

in general, we have a nonlinear opt. problem

1. use nonlinear opt solvers
2. use SCP (or SQP, Q: quadratic)

QP: quad cost & linear constraints.

1. start with NL opt. problem
2. come up with an initial guess for solution.
3. 'Convexify the NL opt.
   a. make the cost quadratic (2nd order Taylor expansion about initial guess)
   b. make the constraints linear (1st order Taylor expansion about initial guess)
4. Solve QP (solution becomes "initial guess")

https://www.cvxpy.org/index.html, https://github.com/cvxgrp/cvxpygen, https://jump.dev/JuMP.jl/stable/, https://github.com/kevin-tracy/qpax, https://www.mathworks.com/help/optim/ug/fmincon.html

# Potential drawbacks? Advantages?

- convexification - potential challenges in convergence & getting global optimal
- multiagent settings - computational scalability.
- SCP - requires 2nd order, may not be available or
  - can try surrogate models.

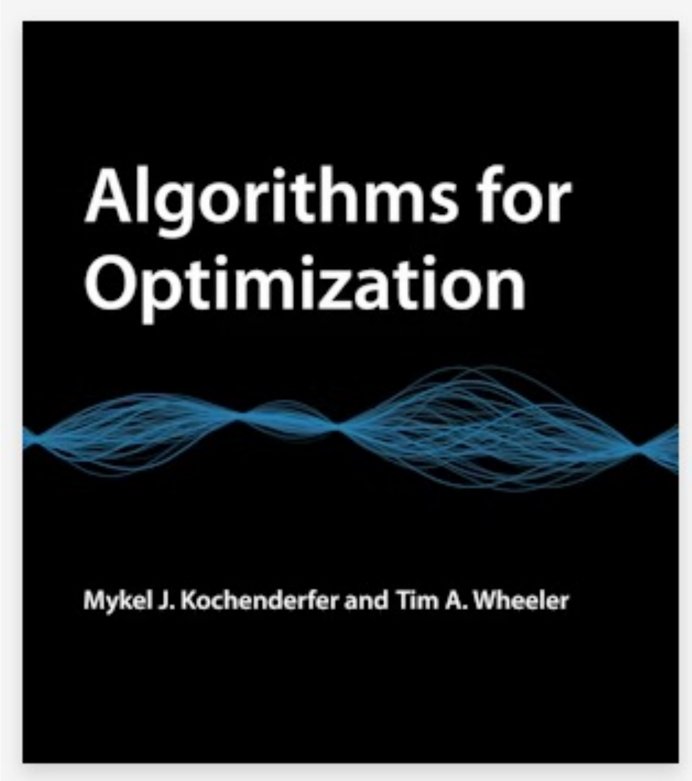
- can't be overly courteous

https://www.cvxpy.org/index.html, https://github.com/cvxgrp/cvxpygen, https://jump.dev/JuMP.jl/stable/, https://github.com/kevin-tracy/qpax, https://www.mathworks.com/help/optim/ug/fmincon.html

# What if we can’t compute gradients easily?

- So far, the methods relied on some sort of gradient descent.
- What if we can’t compute gradients easily?
- We can consider searching over the space via a sampling-based approach
  - Leverage computation!

# Other ways to solve an optimization problem




https://algorithmsbook.com/optimization/files/optimization.pdf

Cross entropy method