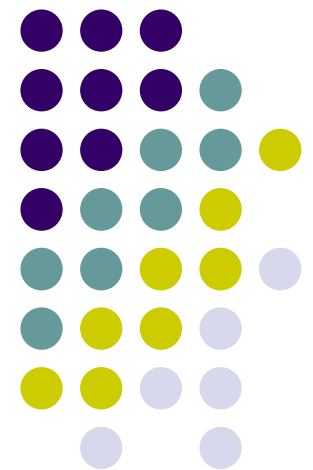


# CS 525M Mobile and Ubiquitous Computing Focus on Projects

---

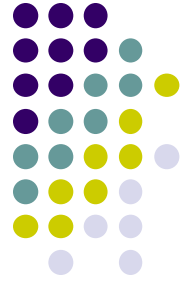
**Emmanuel Agu**





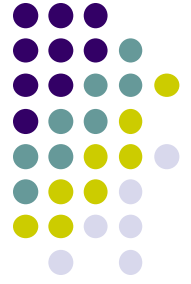
# The Problem

- Problem must have societal benefit
  - If project succeeds, *clearly* saves lives, money, time, etc.
  - Example: Project helps with health problems, education, organization
  - Entertainment (games, etc)?
    - Too many different tastes. Some people don't care
    - Game that's fun to one person, annoying to another



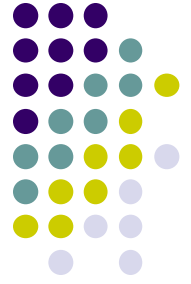
# Problem Questions

- Proof that problem exists
- Cite statistics, numbers from papers. E.g
  - 7.8 percent of Americans have type 2 diabetes
  - Diabetes costs \$178 billion annually to system
  - \$78 billion is wasted in traffic every year
- Why? Big problem = potentially big savings
- If no papers with numbers, do pre-survey to show
  - problem exists
  - People would like/use your solution



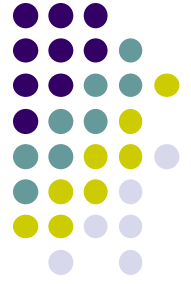
# Methodology

- This is a **graduate** computer **science** class
- Graduate: You have already taken intro classes
- Invalid excuses:
  - I can't program
  - I don't understand operating systems
- Computer **Science** class
  - Science: Systematically generating *knowledge* NOT building products
  - Use scientific method



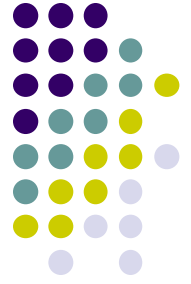
# Projects Types

- **Science:** About knowledge generation
- **Engineering:** building things to solve problems
- Scientific method
  - Construct hypotheses (what do you believe)
    - E.g. Locations with bad signal strength have poor throughput
  - Develop experiments to prove/disprove hypothesis
    - E.g measure throughput at locations with bad sig. strength
  - Experiments must be **reproducible**



# Engineering Projects

- Can build things to solve problems
- Make reasonable assumptions
- Decompose problem into parts, Solve each piece with reasonable tools
- CrowdSense paper Example:
  - Sphynx speech analysis engine (open source)
  - OCR recognition module from microsoft
  - Object recognition
  - Indoor scene classification



# Engineering Projects

- Must evaluate! Evaluate! Evaluate!
  - Demonstrate that solution presented works either quantitatively or through user studies
  - Under what conditions your system works well/not?
  - That's why papers present so many graphs
  - Start with nice design and justify choices
    - Think: have a sub-problem = what component solves that?
    - E.g. Require fast access to lots of data = hash tables



# Engineering Projects

- More difficult to do projects not leveraging some of work in previous papers
- Don't want shooting from hip
  - I like this project so I'm doing it
  - Build what's cool to you
  - Don't use scientific method or good engineering
  - Don't evaluate to prove it works