

Camfora: oblivious robot coordination simulator

- Liang Nan Dong

- What is this?
 - Simulation of **mobile, autonomous robots** and how they can **coordinate** even if they **can't communicate** and **don't remember anything**.
- Why is this interesting?
 - Theory of computation: can we do it?
 - Mobile geometry
 - Practical considerations: do more with less
 - It's fun!

Contents

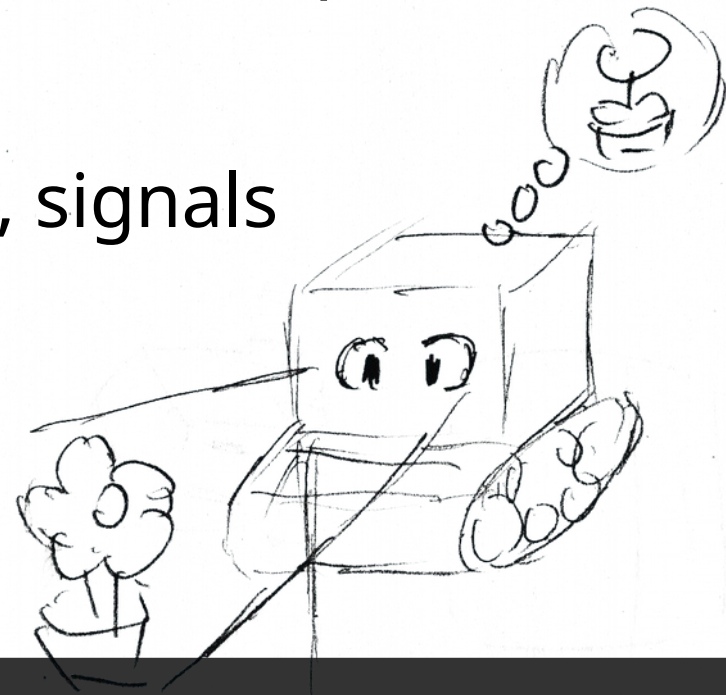
- The mobile robots problem
- Previous work
- Motivation
- The Camfora simulator
- Discussion

Autonomous humans

- Think of a person, trying to go somewhere, they have to do these things:
 - See/hear/touch surroundings
 - Understand what they're seeing/hearing
 - Think and decide what to do
 - Act on the decision

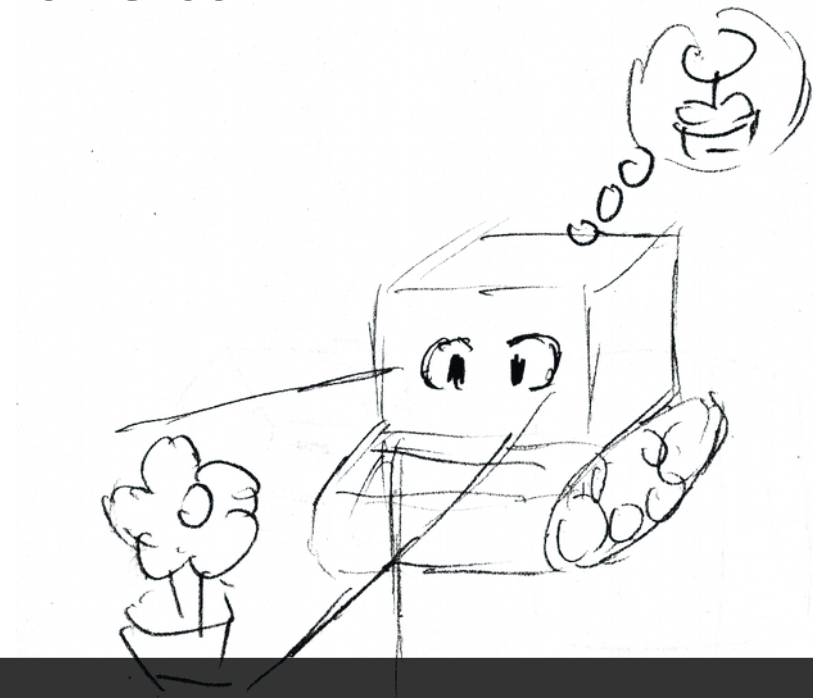
Autonomous robot

- An autonomous robot also has to be able to do these things:
 - **Sensing:** like camera, lidar, compass, gyro, GPS...
 - **Recognition:** understand it's a car, it's a person
 - **Computation**
 - **Output:** adjust power, steering, signals



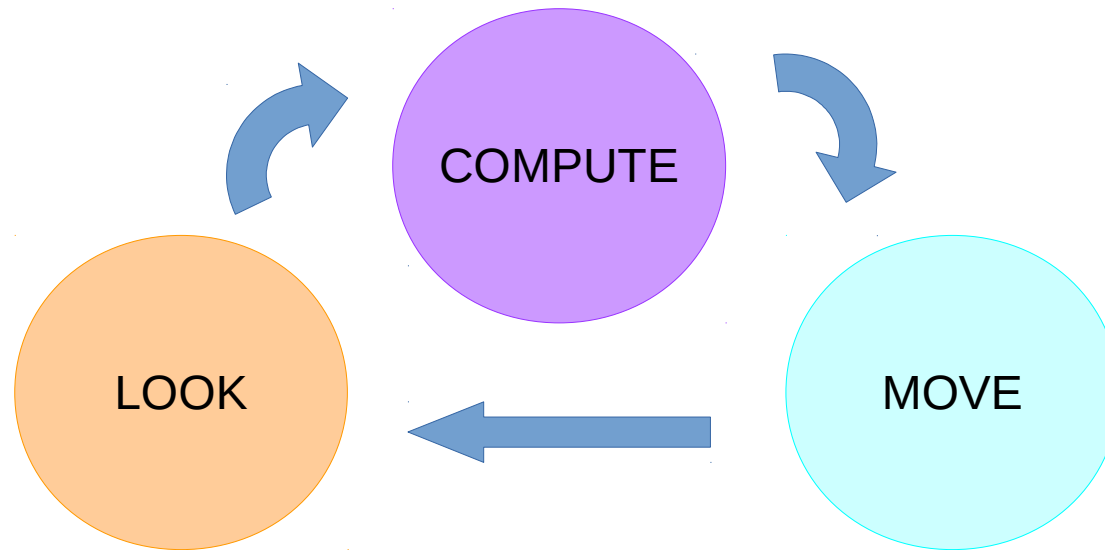
Mobile robot model

- In our theoretical model, it's simplified:
 - *LOOK*: get information from surroundings
 - (understanding is assumed to be perfect)
 - *COMPUTE*: decide a goal to move to
 - *MOVE* to the decided point



Mobile robot model

- In this model they run in stages



- Additionally,
 - Robots are sizeless
 - And oblivious (have no memory)

Variations - capabilities

- Different conditions the robots are under
- Including how much resources they have
 - Coordinate systems: (dis)agreement on distance, orientation
 - Vision: range, obstruction, multiplicity detection
 - Memory: oblivious, constant, limited, unlimited
 - Movement: rigid, minimal/maximal distance
 - Communications: lights / no lights
 - Power: unlimited/limited
 - and so on...



Variations – disturbances

- Faults
- Add/remove
- Self-stablization

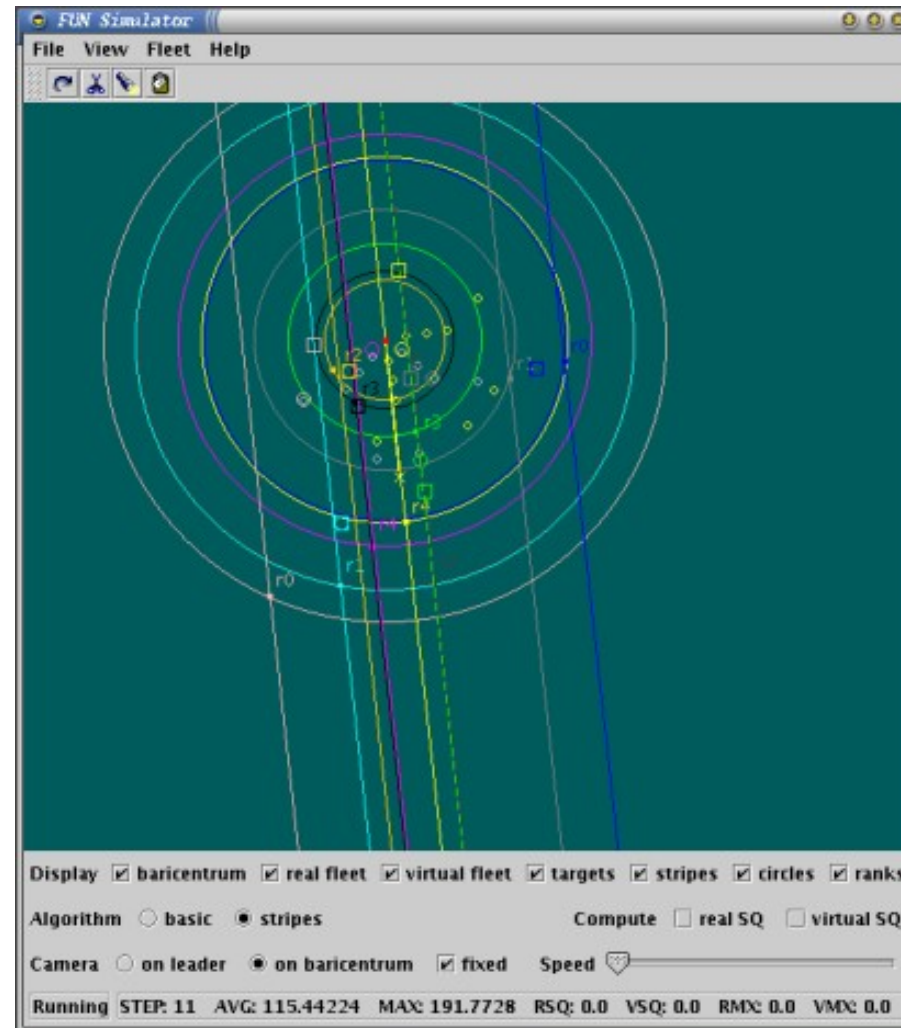
Variations – time

- Robots also run on a schedule
- Usually thought as controlled by a central **“scheduler”**
- (something about wi-fi High Performance Browser Networking Ilya Grigorik)

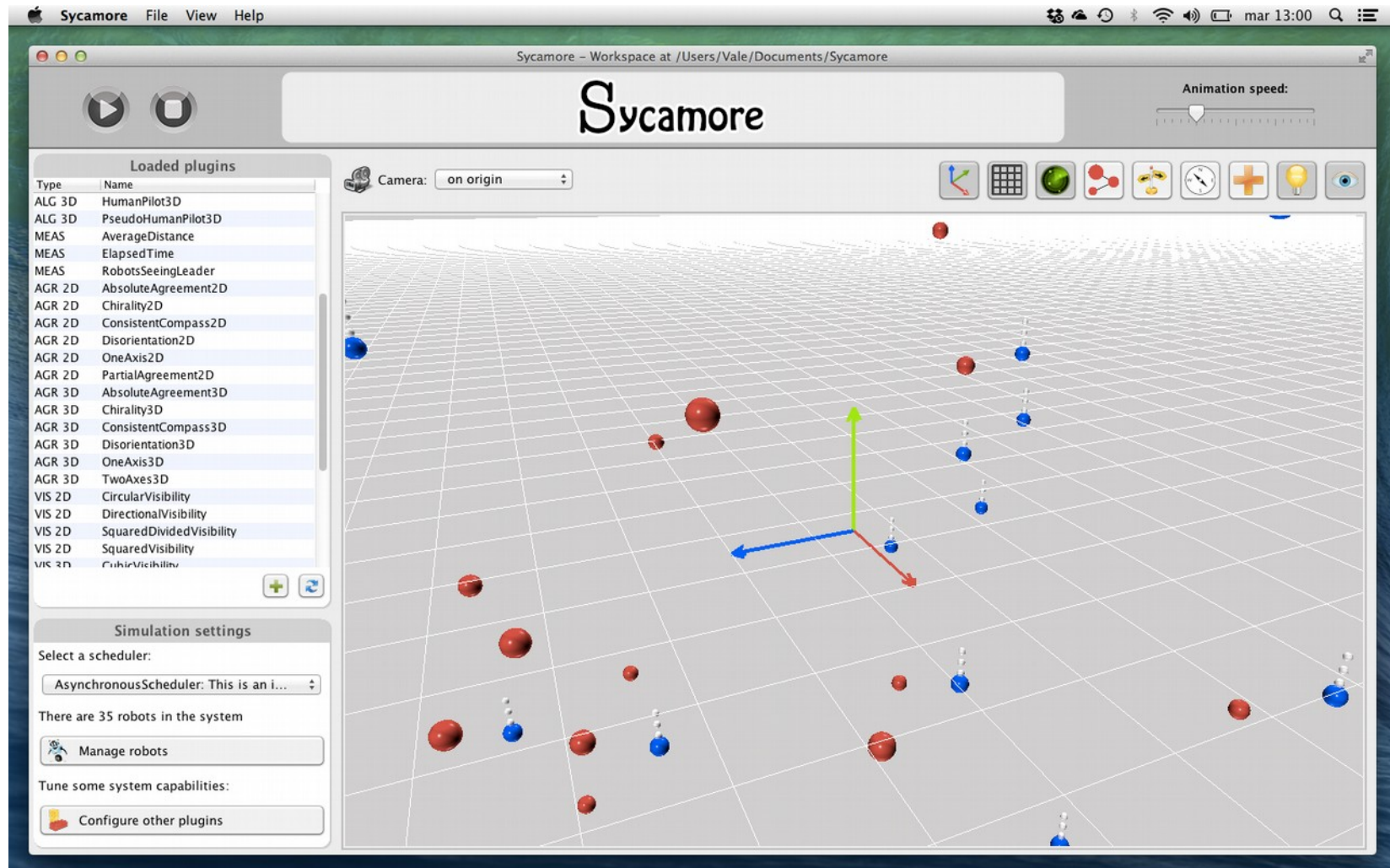
Contents

- The mobile robots problem
- Previous work
- Motivation
- The Camfora simulator
- Discussion

CORDA simulator



Sycamore



Contents

- The mobile robots problem
- Previous work
- Motivation
- The Camfora simulator
- Discussion

Motivation

- Sycamore is great, but...
 - You need to download Java to run it, and also need the JDK to build on it
 - It isn't interactive
- Want something that
 - Easier to run and develop
 - Is interactive
 - Is extensible

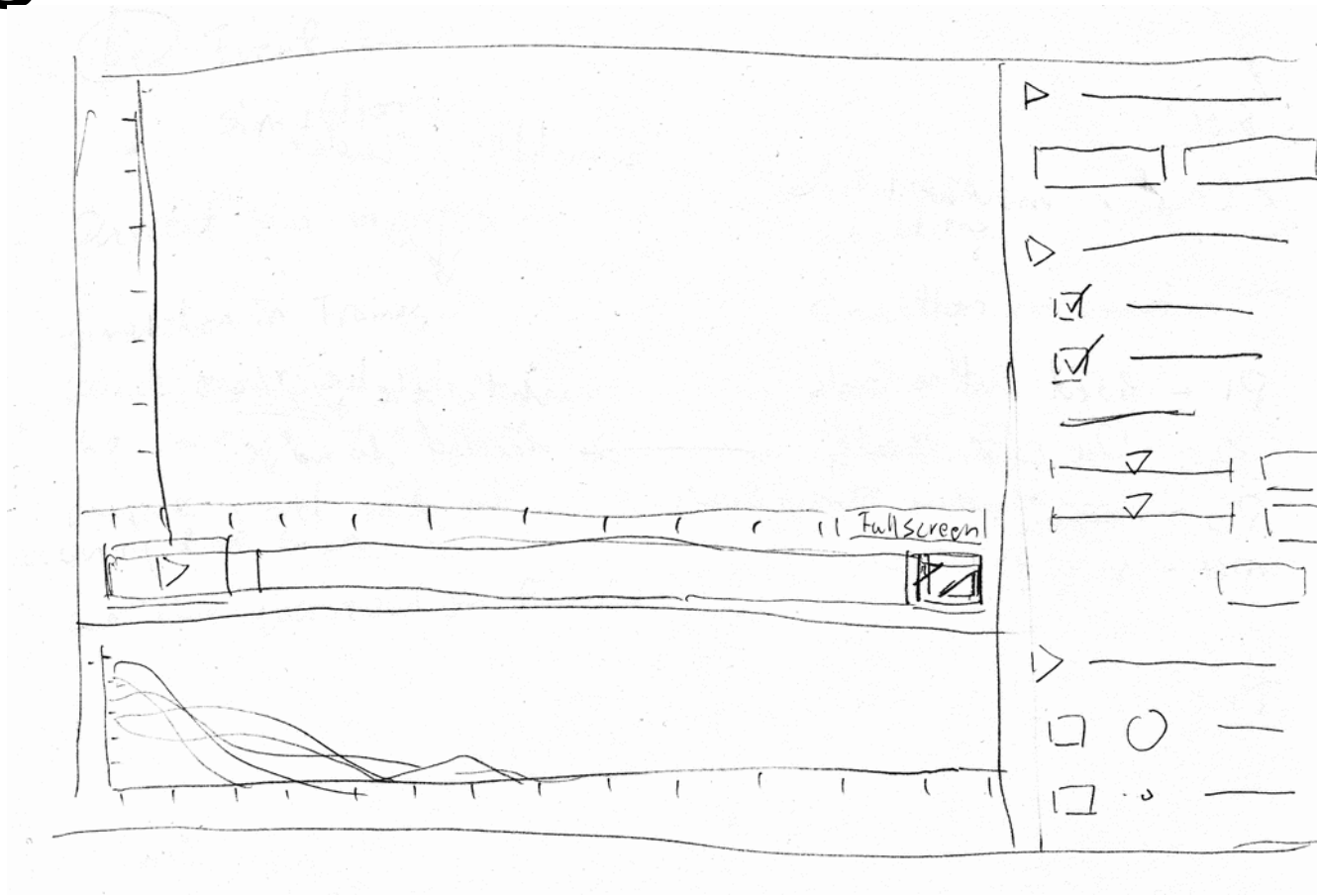
Motivation (cont.)

- Rich Internet Application (fat client)
- Wants something that
 - Fast to download
 - Smooth animations
 - (running outside the browser would be nice, too)

Contents

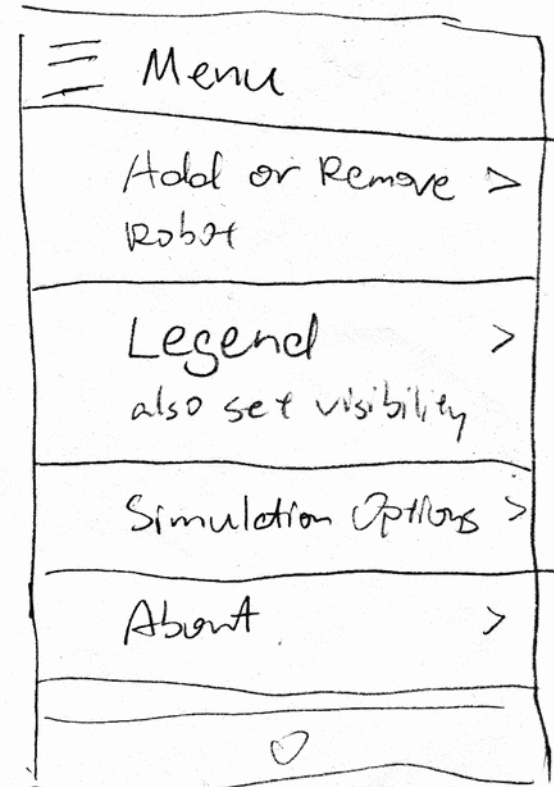
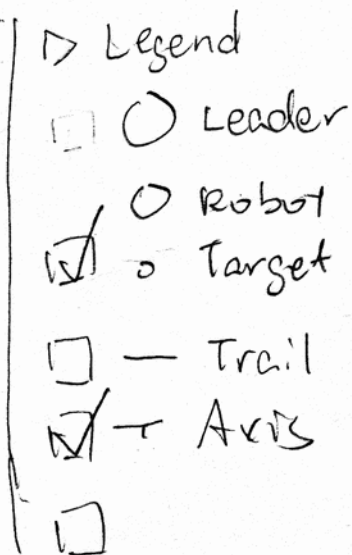
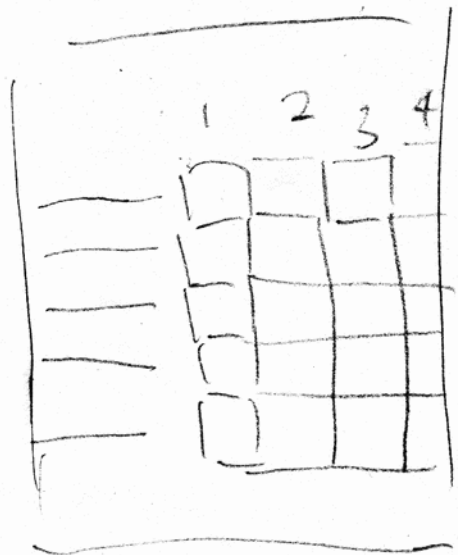
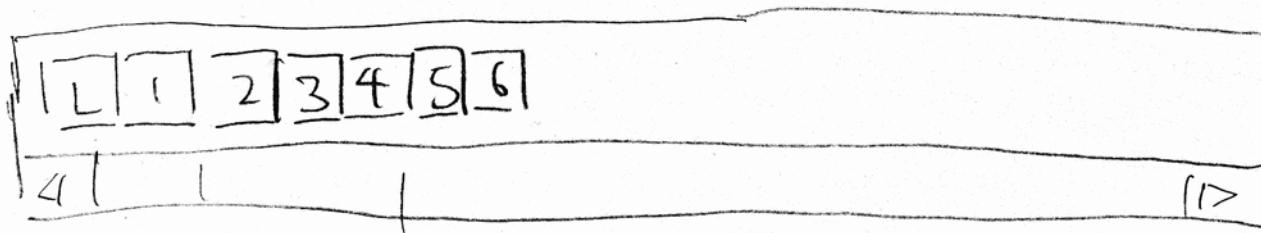
- The mobile robots problem
- Previous work
- Motivation
- The Camfora simulator
- Discussion

Design: UI



- Embed in web page, or take full window
- Use sidebar for commands

Design: UI



- Sidebar collapses on mobile
- Exploration of command palette

Design: Visualization

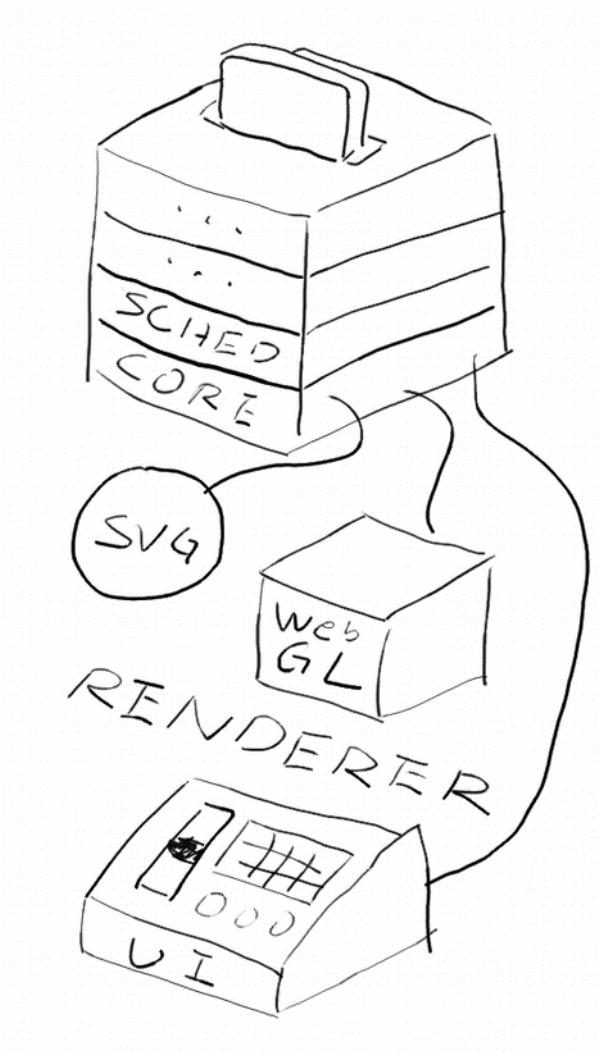
- Visualizing movement:
 - Axes, trails
- Goals
- Algorithm specific visualization: to be implemented separately.

Implementation

- Using JavaScript
- Dependencies:
 - D3 : visualization library, lots of useful tools
 - Glmatrix: vector mathematics without creating lots of temporary objects
 - Rollup: compile JavaScript-of-tonight modules into JavaScript-of-today
- Using ES6 modules, rest is ES5
- Tries to avoid temporary objects every frame (not always successful)

Implementation – Architecture

- Model (simulation)
- View (renderer & UI)
 - Currently: Scalable Vector Graphic
 - Decoupled renderer: SVG, HTML+CSS, OpenGL...
 - Or running without graphics
- Controller (environment & API)
 - Drives simulation
 - Allows running fast or slowly



Implementation – Event loop

- Basic event loop:
 - Handle events
 - Update game world
 - Render visuals (and audio)
 - Repeat

Inside the update function

- Update simulation time
- Calculate new positions
- Check state changes
- Do LOOK and COMPUTE
 - Plugins as filters on LOOK result
- Render and update UI

Implementation – Event loop

- In browsers you're living in somebody else's event loop.
- So it's important for performance not to write to the DOM during layout
 - (avoid write-read hazard)
- Using `RequestAnimationFrame` instead of `SetInterval` / `setTimeout` for more efficient animation

Demo

Contents

- The mobile robots problem
- Previous work
- Motivation
- The Camfora simulator
- Discussion

Discussion

- Turns out avoiding Java doesn't entirely avoid problems arising with complexity
 - Once it gets complex, browser no longer suitable as Integrated Development Environment
 - Still depend on libraries, build-tools
- JavaScript modules and packages is confusing

Future Work

References

- Gervasi, V. and Prencipe, G., 2004. Coordination without communication: the case of the flocking problem. *Discrete Applied Mathematics*, 144(3), pp.324-344.
- Gervasi, V. and Prencipe, G., 2003, October. Robotic cops: The intruder problem. In *Systems, Man and Cybernetics, 2003. IEEE International Conference on* (Vol. 3, pp. 2284-2289). IEEE.
- VOLPI, V., 2013. Sycamore-2D/3D Mobile Robots simulation environment.

- Slides:
<http://kakuradystatic.nfshost.com/files/2016/camforajs/>
- Code:
<https://github.com/Kakurady/camforajs>
- Best wishes on your exams!

