CS/EE/ME 75(c): Organization

- Meeting Time?
  - **General Class Meeting:** Evening, or day time?
  - **Once-per-week team meeting** with Joel/Grad students.
- Documentation:
  - Need to keep weekly presentations on GitLab page.
  - Need to document all results for end-of-quarter
  - Gant/Pert charts & Schedules

CS/EE/ME 75(c): Key Issues and Questions

- Time to complete and integrate!
- Teaming Structure. Keep or change?
    - Battery Swap:
    - Drive-o-Copter
    - Clutch-o-copter
    - Ground Vehicles

- Refine/Update goals

CS/EE/ME 75(c): Needs

- Battery Team:
  - Mechanical Prototype of Battery Swap Robot Mechanism
  - Mechanical Prototype of Battery Swap Structure
  - ConOps—Concept of Operations

- Drive-o-Copter
  - Complete flying control
  - “Tune” flight control system
  - Characterize flight performance and flight time
  - Weight Reduction Analysis
  - Complete “Tread-o-copter”
  - Characterize ground performance with different chasees:
    - 4-wheel skids steer, 6-motor Ackerman, swerve steer, treads/tracks

CS/EE/ME 75(c): Needs

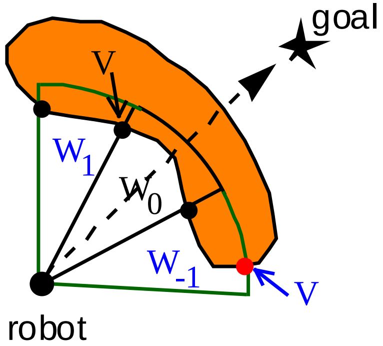
- Clutch-o-copter:
  - Bench-top mechanical Prototype of Clutch/shift Mechanism
  - Characterize device
  - Decision Point: build prototype, or join another effort
- Ground Vehicle: hardware
  - Next generation of “Automation tower”
  - Automate RC Car (mechanical and control system)
    - Connection servos and ESC to NUC.
    - “tune” the steering and speed control systems
- Ground Vehicle: Algorithms/Software
  - Efficient planning of dynamic vehicle motions in rough terrain

- Efficient planning of dynamic vehicle motions in rough terrain
  - Efficient ways to create a model of the terrain from Lidar/Real Sense point cloud data—survey and test ideas of others
    - GP occupancy grids, elevation maps, better cost maps
    - None of them are very efficient…….
  - We still haven't gotten to the core of the problem:
    - how to plan a vehicle trajectory that most rapidly moves toward a goal, or covers a given area
    - Dynamic optimization methods
  - **New idea:** only focus on the data you need, and therefore only convert a small portion of sensor data to a detailed map. Even
    - ***Wedgebug?!***

Wedgebug

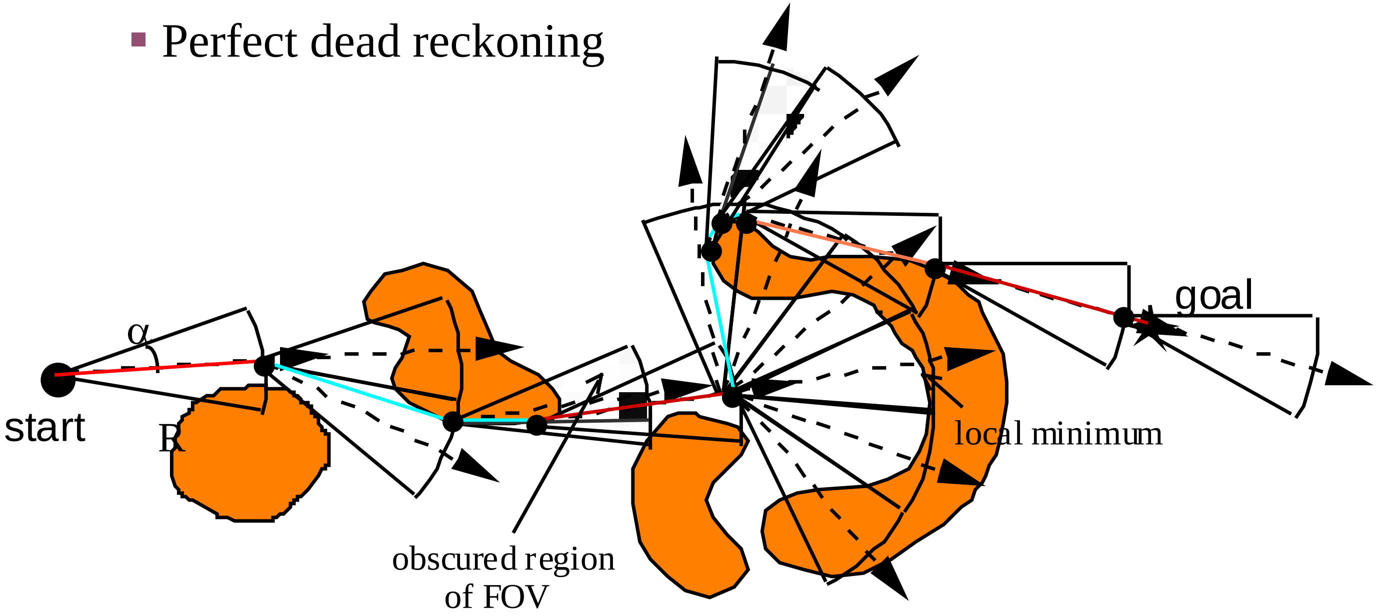
(Laubach & Burdick 1999)

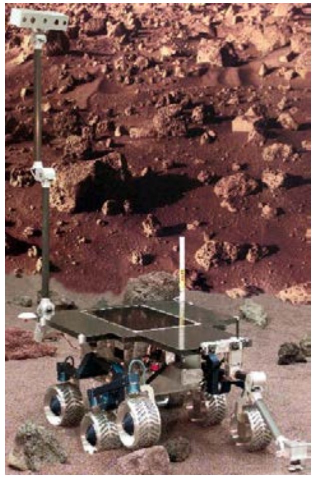
- Based on *Tangent Bug*
- Minimizes local path length of robot travel
- Minimizes (locally) the amount of sensing



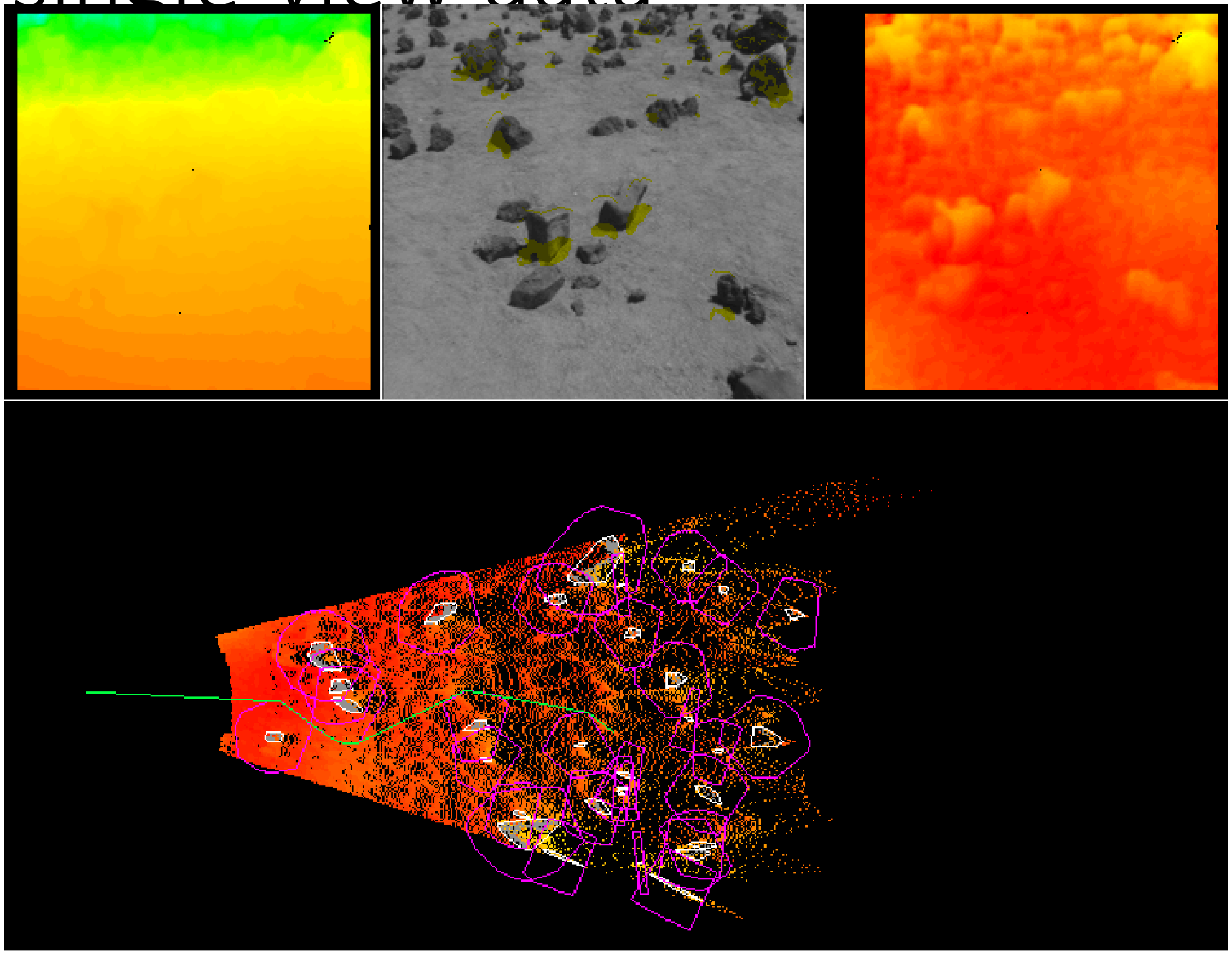
Assumes:

- Range sensed in a "wedge shaped" region
- Perfect dead reckoning





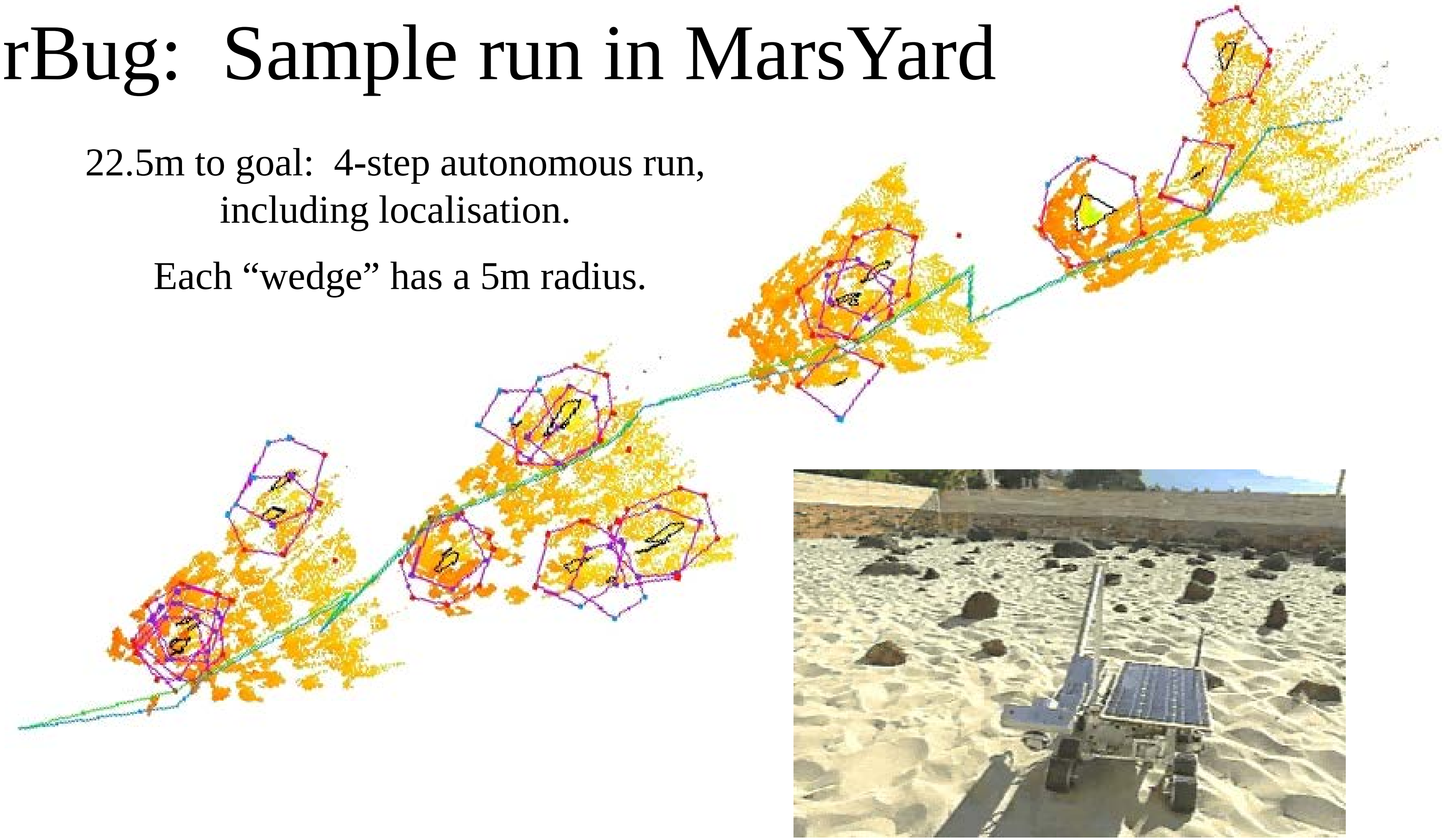
Sample single-view data



RoverBug: Sample run in MarsYard

22.5m to goal: 4-step autonomous run, including localisation.

Each "wedge" has a 5m radius.



The "jogs" in the path after each wedge are not rover motions, but are the results of the visual localization procedure.