**How to REALLY Customize Your Car!!**

- A Nascent RoSES Customization Manager

**Where, oh where, should my software run?**

*Given:*
- A collection of microcontrollers, each with a vector of discrete resources.
- A software flow graph, where functionality nodes have resource requirements & arcs have dataflow requirements.

*Do:*
- Pack the nodes -- starting with “largest” -- onto the microcontroller that will minimize network use.
- Watch for size or other constraints!!
- Continue until all nodes pack or they overflow available microcontrollers.

**How many embedded processors do I need?**

- Start with minimal set of small microcontrollers.
- Keep packing tasks until failure.
- At each failure, decide to either grow a microcontroller’s specification or add another processor.
- Make decision based on cost model and effect on network bandwidth.

**What if I want certain sensor & actuator software to co-exist? Or not?**

Cluster during a pre-packing phase.
Exclude when checking packing constraints.
**BONUS - Algorithm runs faster!**

**Where will this research go from here?**

*Customization manager for RoSES will:*
- Optimize functionality system-wide for available hardware
- Choose algorithms/adapters for installation on sensors & actuators
- Abide by real-time scheduling constraints, both in the CPU & network

*GOALS:*
- Graceful degradation
- Graceful upgrade
- Product family architecture design
- Logistic benefits
- Replacement with non-exact spares
- Reduced need for legacy parts