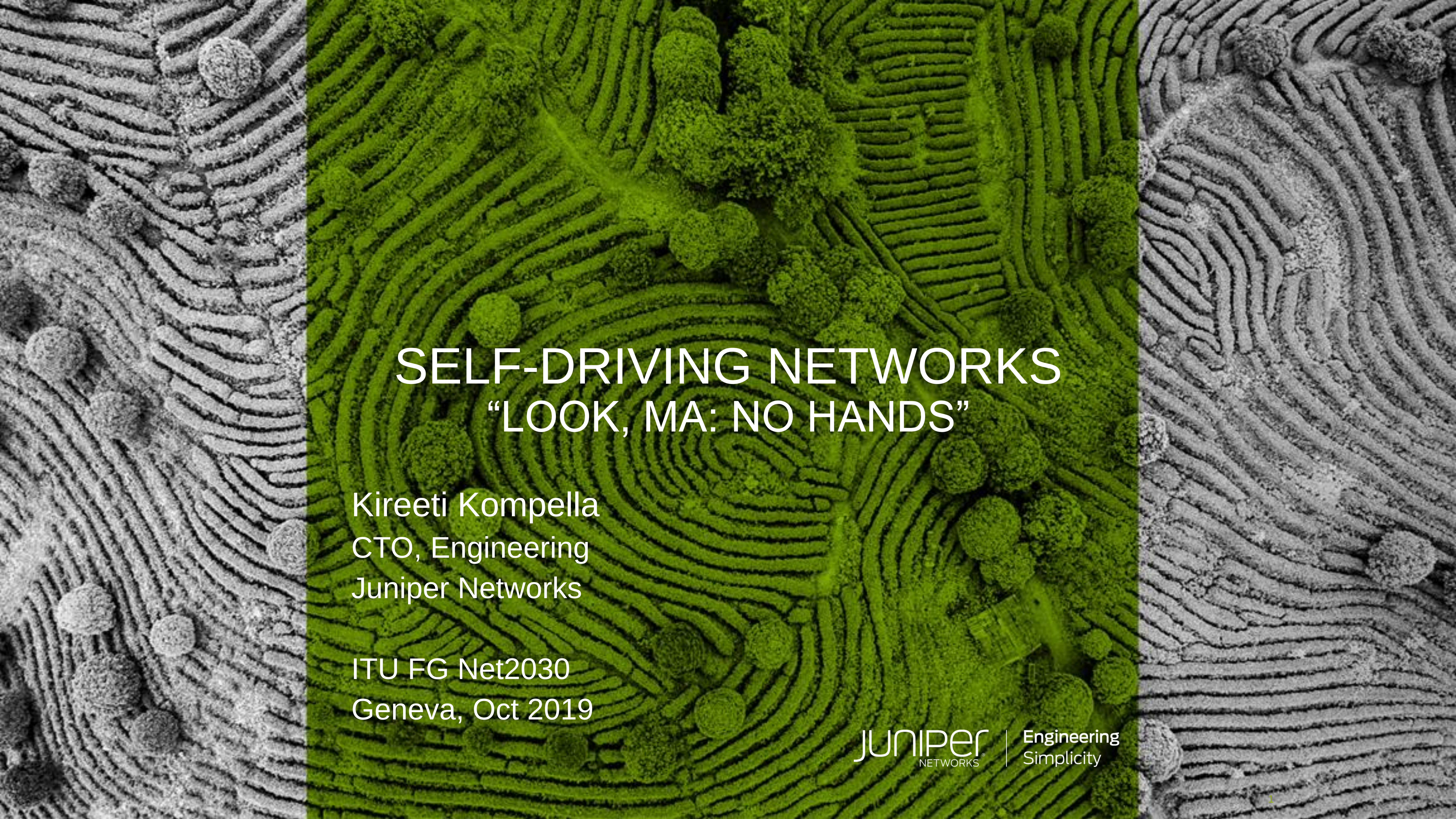


SELF-DRIVING NETWORKS
“LOOK, MA: NO HANDS”

Kireeti Kompella
CTO, Engineering
Juniper Networks

ITU FG Net2030
Geneva, Oct 2019

JUNIPER NETWORKS | Engineering Simplicity

VISION

JUNIPER NETWORKS | Engineering Simplicity

THE DARPA **GRAND CHALLENGE**

BUILD A FULLY AUTONOMOUS GROUND VEHICLE

GOAL

Drive a pre-defined 240km course in the Mojave Desert along freeway I-15

PRIZE

$1 Million

RESULT

2004: **Fail** (best was less than 12km!)
2005: 5/23 completed it

2007: "URBAN CHALLENGE"

Drive a 96km urban course following traffic regulations & dealing with other cars
6 cars completed this

IMPACT

- **Programmers, not drivers**
- No cops, lawyers, witnesses
- Quadruple highway capacity
- Glitches, insurance?
- Ethical Self-Driving Cars?

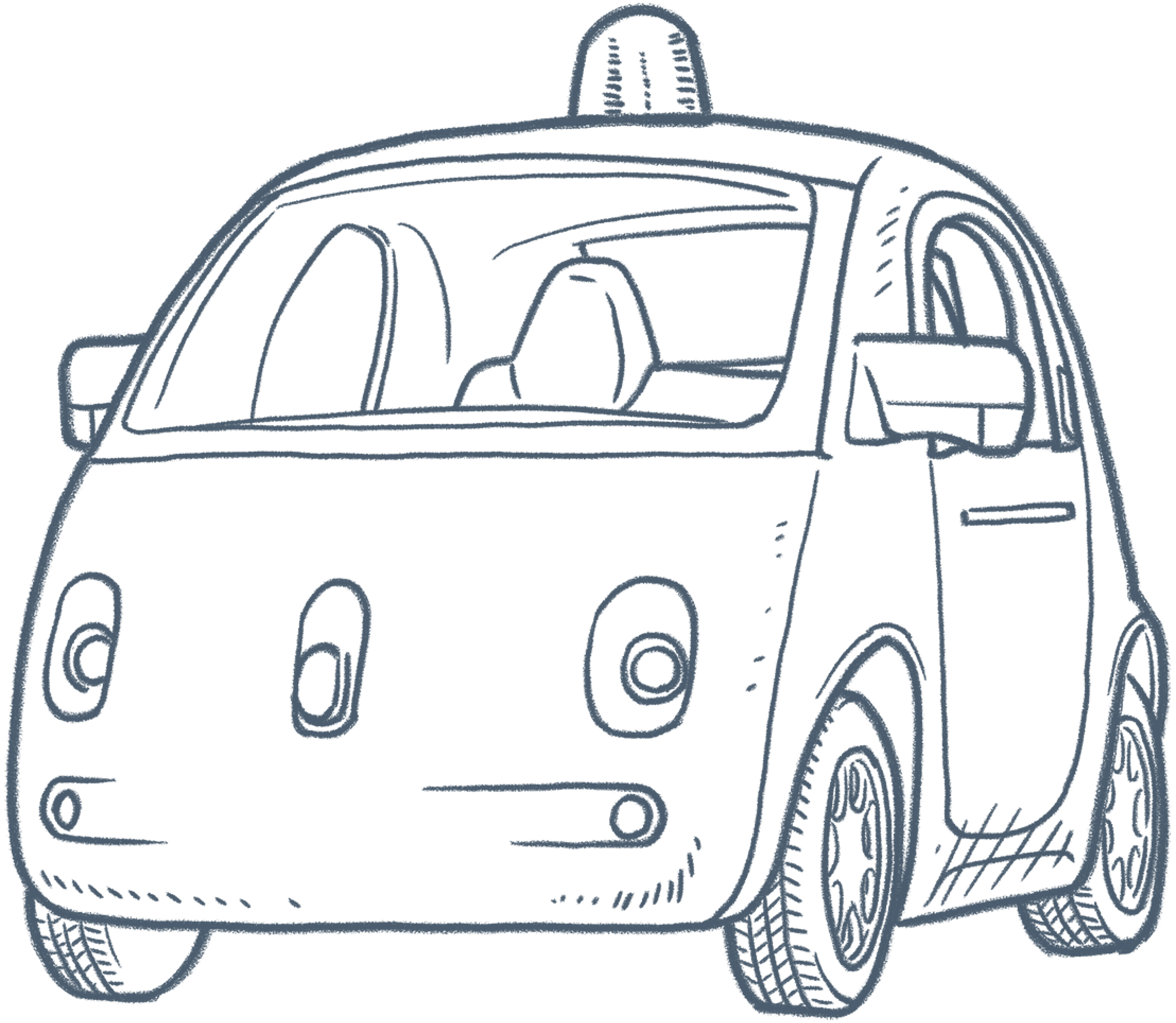
POSSIBILITIES

GRAND RESULT: THE SELF-DRIVING CAR: (2009, 2014)

No steering wheel, no pedals—
a completely autonomous car

Not just an incremental improvement

This is a **DISRUPTIVE** change in automotive technology!



THE NETWORKING **GRAND CHALLENGE**

BUILD A SELF-DRIVING NETWORK

GOAL

Self-Discover—Self-Configure—Self-Monitor—Self-Correct—Auto-Detect Customers—Auto-Provision—Self-Diagnose—Self-Optimize—Self-Report

RESULT

Free up people to work at a higher-level: new service design and "mash-ups"

Agile, even **anticipatory** service creation

Fast, intelligent response to security breaches

CHALLENGE

Build and operate a self-driving **service network** that **greatly increases agility** and **vastly improves service quality** by **proactive maintenance**

Autonomously run the end-to-end life-cycle of a service

Learn user behavior and anticipate changing user requirements

IMPACT

- **New skill sets required**
- **New focus**
  - BGP/IGP policies → AI policy
  - Service config → service design
  - Reactive → proactive
  - Firewall rules → anomaly detection

POSSIBILITIES

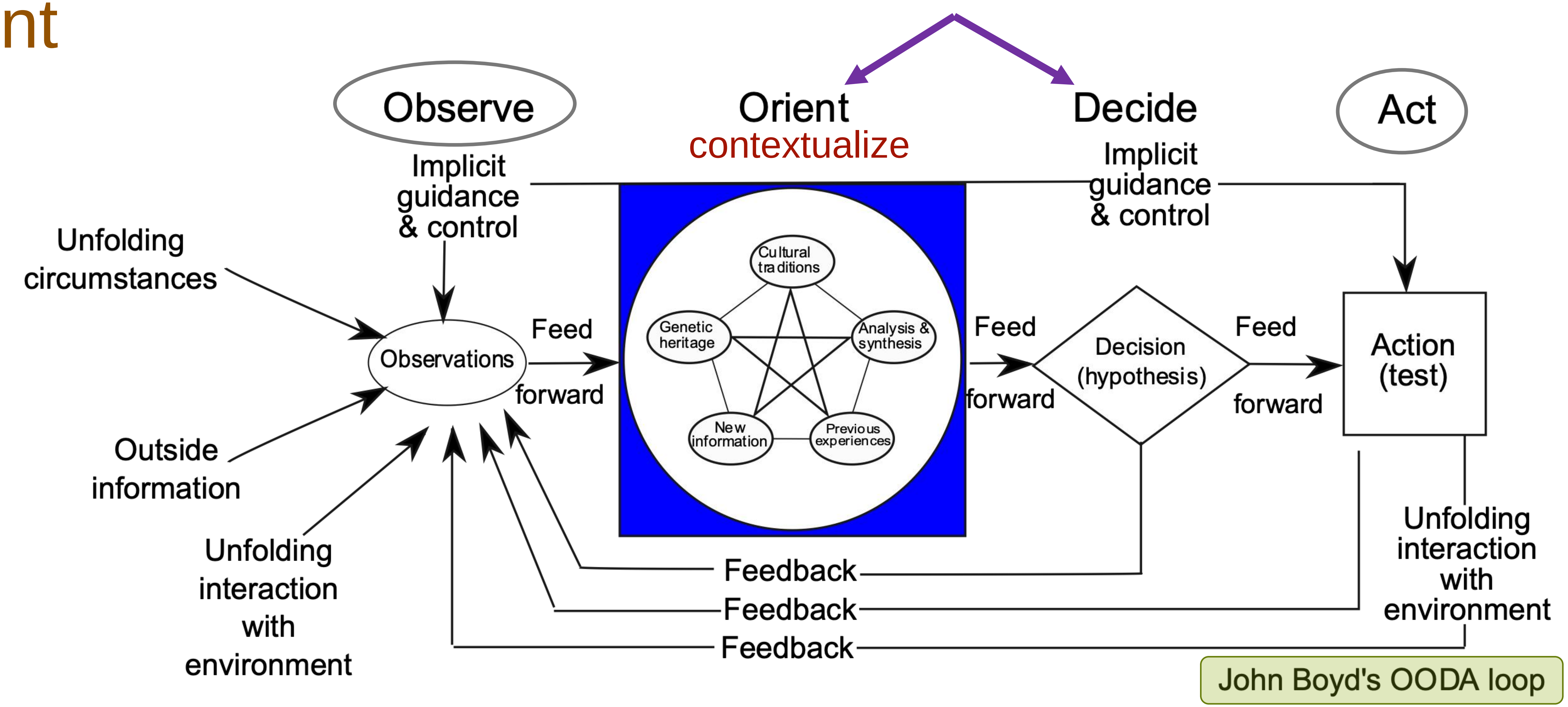
CONCEPT

What would this look like?

Think **SON**, but applied to all aspects of operations, to all parts of the network

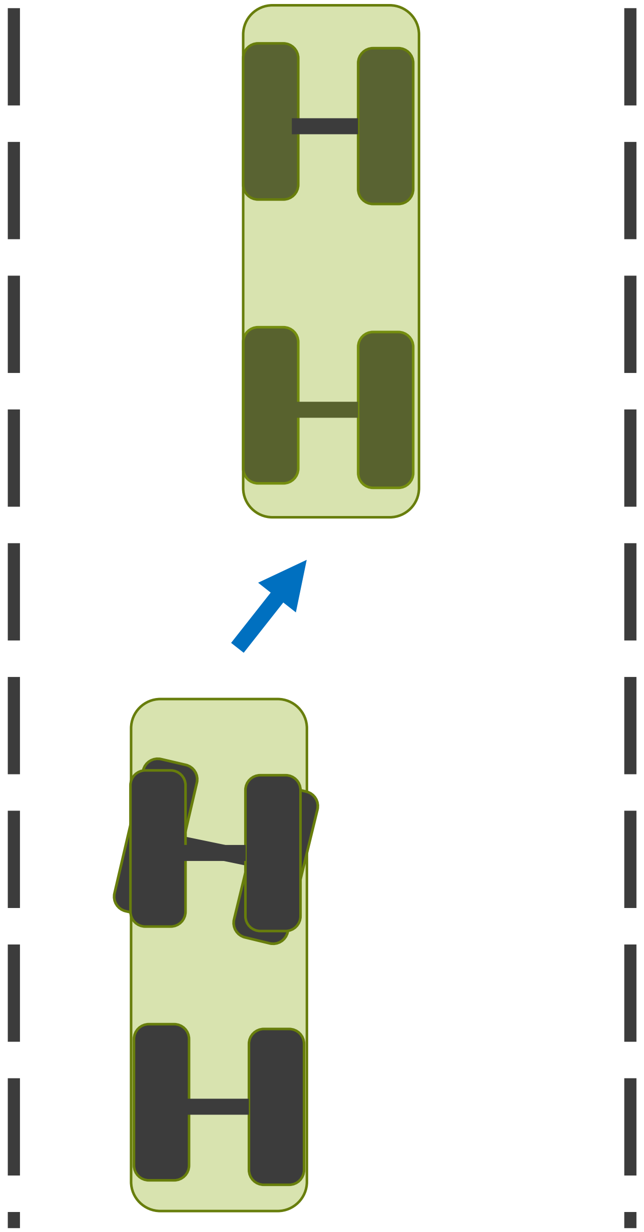
CLOSED LOOP SYSTEMS (THE OODA FRAMEWORK)

Intent



BREAK DOWN THE PROBLEM TO MANAGEABLE CHUNKS

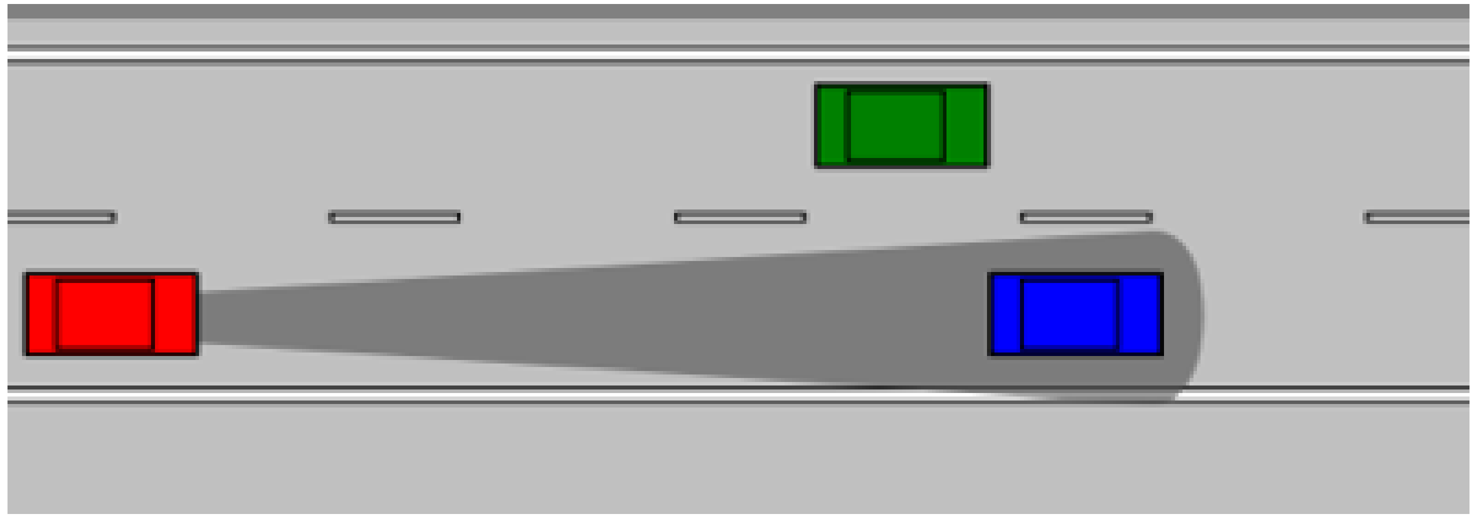
ILLUSTRATION: AUTOMATIC LANE CENTERING

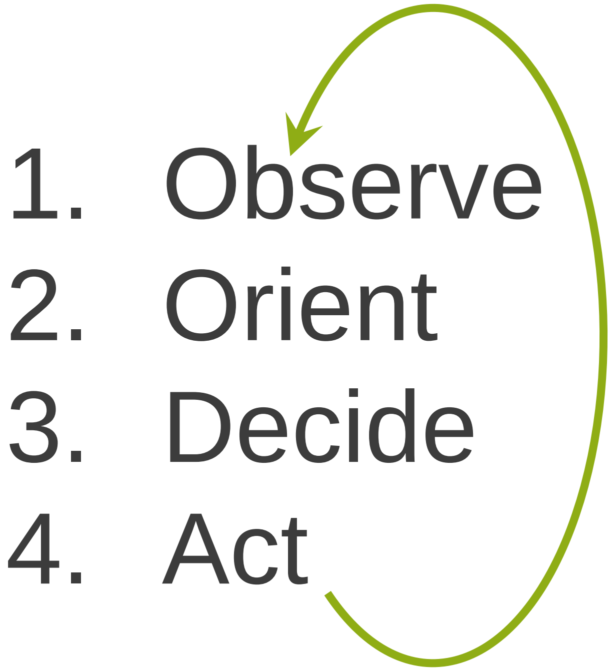


Intent

1. Observe
2. Orient
3. Decide
4. Act

ILLUSTRATION: AUTOMATIC SPEED CONTROL





Intent

1. Observe
2. Orient
3. Decide
4. Act

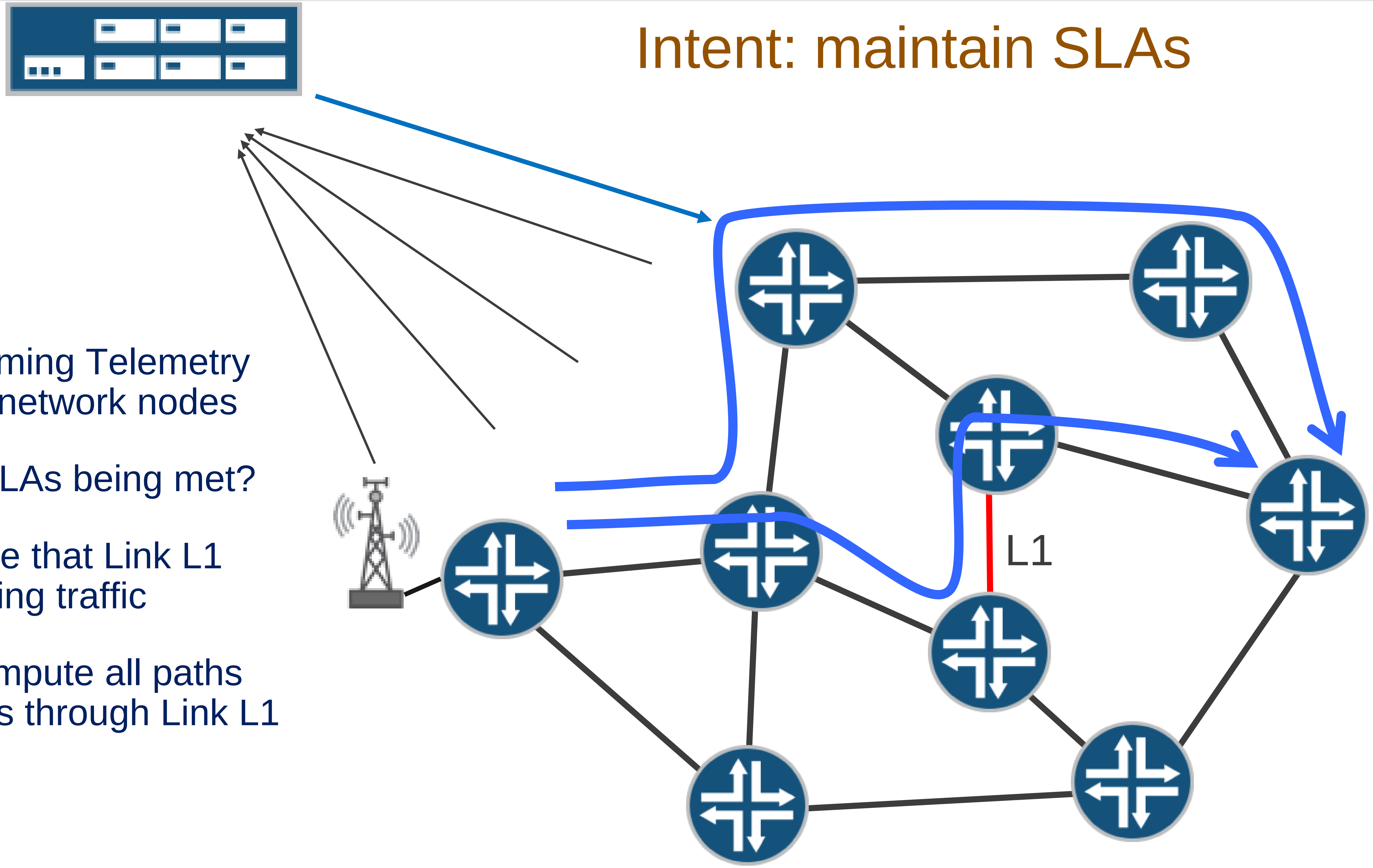
Act: while braking, must stay within lane

Act: may decide to change lanes instead of (or even while) braking

As subsystems interact, you get more complex and more capable systems

ILLUSTRATION: SELF-MANAGED SLAs

Intent: maintain SLAs



1. Streaming Telemetry from all network nodes

2. Are SLAs being met?

3. Decide that Link L1 is dropping traffic

4. Recompute all paths that pass through Link L1

Simple example, but consider the following:

1. Need this for real-time operation!
2. SLAs getting more complex, critical
3. Decisions must address **root cause**
4. Action must not violate other SLAs

BREAKING THIS DOWN

Architecture

1. Observe: build a robust data ingest engine

All kinds of data: stats, traps, syslogs, thresholds, …. and expertise and experience

2. Orient: understand how the data fits together

Needs deep knowledge of implementation, debugging, NOC expertise & experience

ML

3. Decide: does the behavior match the intent?

Needs deep knowledge of the service, SLAs, deployment, expertise and experience

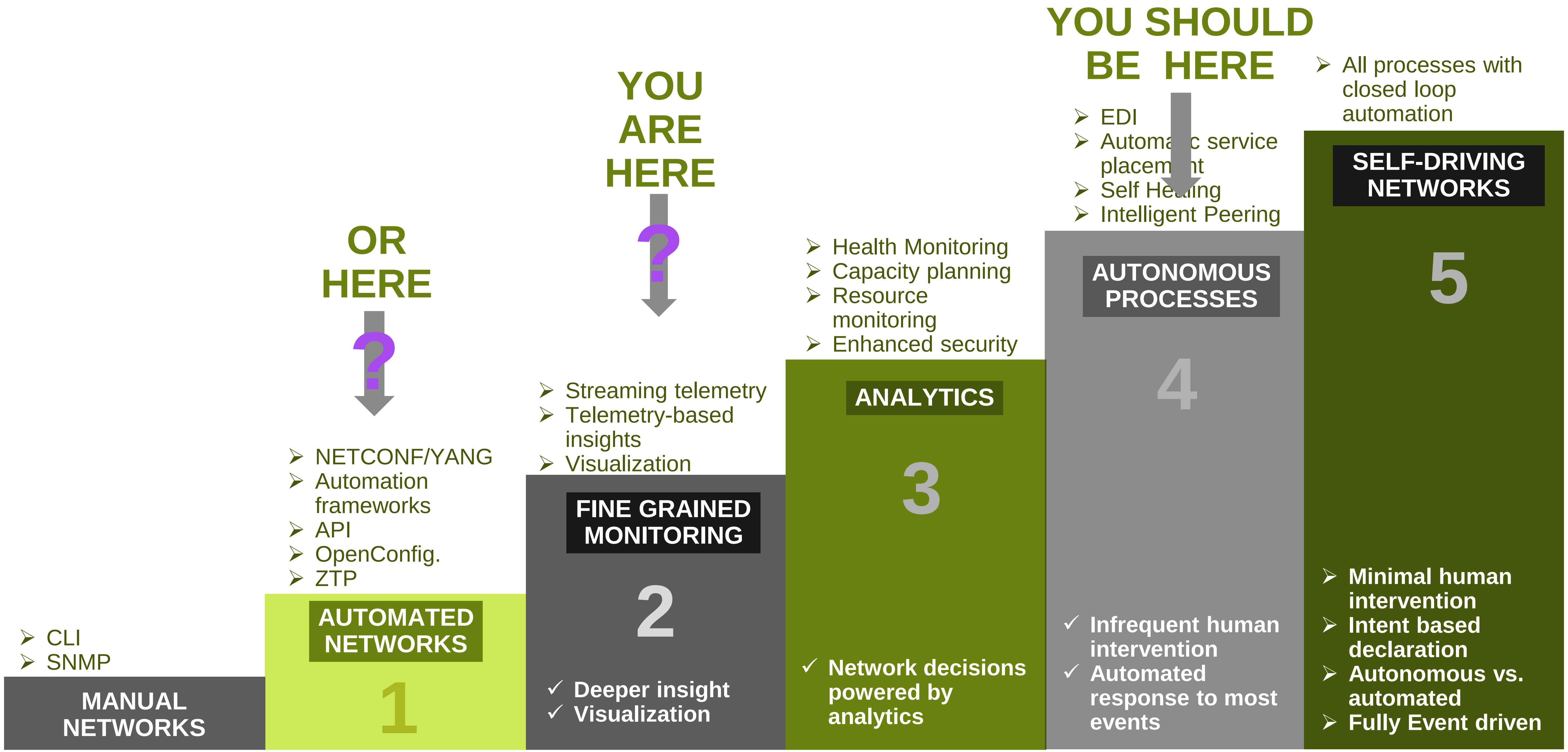
ML

4. Act: return behavior to the intent (via netconf, etc.)

Needs clear understanding of consequences, careful sequencing of steps, expertise and experience

SELF-DRIVING NETWORKS

FIVE STAGES TOWARDS THE LONG-TERM VISION



- CLI
- SNMP

MANUAL NETWORKS

OR HERE ?

- NETCONF/YANG
- Automation frameworks
- API
- OpenConfig.
- ZTP

AUTOMATED NETWORKS

1

YOU ARE HERE ?

- Streaming telemetry
- Telemetry-based insights
- Visualization

FINE GRAINED MONITORING

2

- ✓ Deeper insight
- ✓ Visualization

- Health Monitoring
- Capacity planning
- Resource monitoring
- Enhanced security

ANALYTICS

3

- ✓ Network decisions powered by analytics

YOU SHOULD BE HERE

- EDI
- Automatic service placement
- Self Healing
- Intelligent Peering

AUTONOMOUS PROCESSES

4

- ✓ Infrequent human intervention
- ✓ Automated response to most events

- All processes with closed loop automation

SELF-DRIVING NETWORKS

5

- Minimal human intervention
- Intent based declaration
- Autonomous vs. automated
- Fully Event driven

DEPLOYMENT

What does it take to put this in action?

Engineering
Simplicity

PERSPECTIVE ON CONTROL AND TRUST

MIT Technology Review,

Will Knight, Oct 2013

“Driverless Cars Are Further Away Than You Think”

Musk, Dec 2015: full autonomy by end of 2018

Musk, Dec 2017: full self-driving by end of 2019

GIVING UP CONTROL WORRIES SOME, EXCITES OTHERS

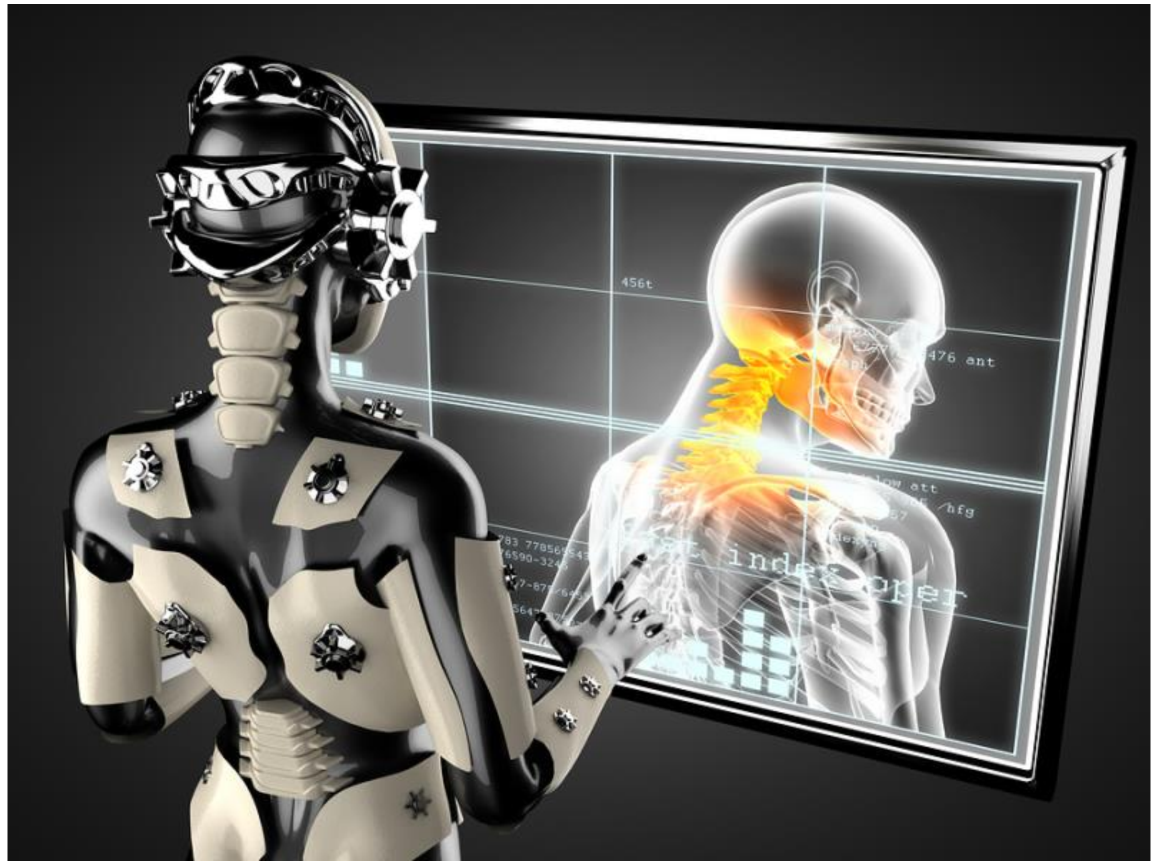


Photo: www.dotmed.com

Photo: www.cio.com.au

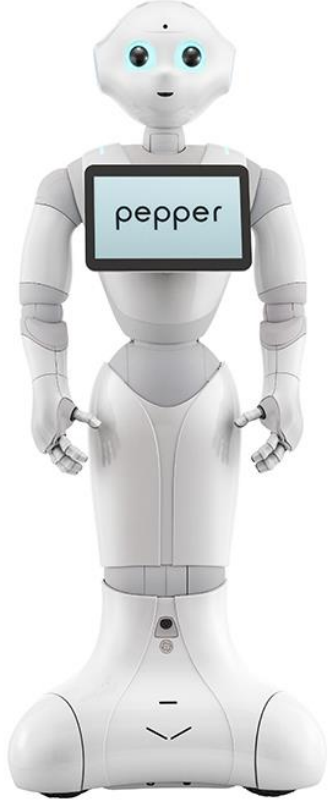


Photo: SoftBank/
Aldebaran Robotics

Photo: newyorker.com

Photo: erobots.in

ML?

The Three Laws of ~~Robotics~~

1. Don’t hurt/kill humans
2. Obey a human’s orders
3. Protect yourself

IF CONTROL COMES FROM ML, UNDERSTAND THIS

Search: one pixel attack

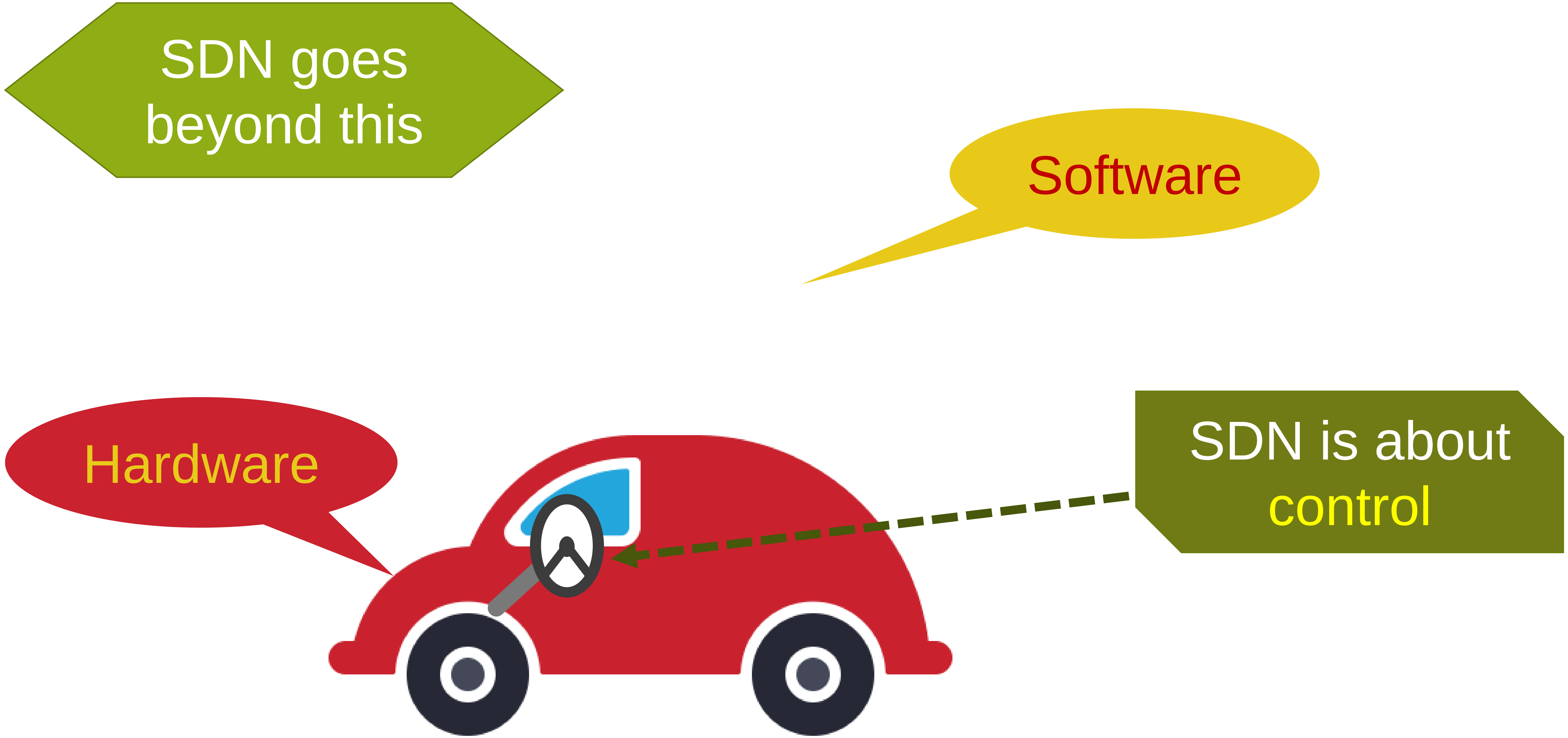
Amazingly accurate in its domains

Limitations

- Opaque & un-understandable — Explainable AI?
- Blind, goal-seeking — Ethical AI?
- No “common sense” — but will that change?
- Easy to fool — adversarial techniques
- Three-year-olds can do much better in most contexts

Pragmatic approach:
rule-based decision making

SOFTWARE-DEFINED NETWORKS: “I WANT MORE CONTROL!”



SDN goes beyond this

Software

Hardware

SDN is about control

MACHINE LEARNING: MAJOR PARADIGM SHIFT

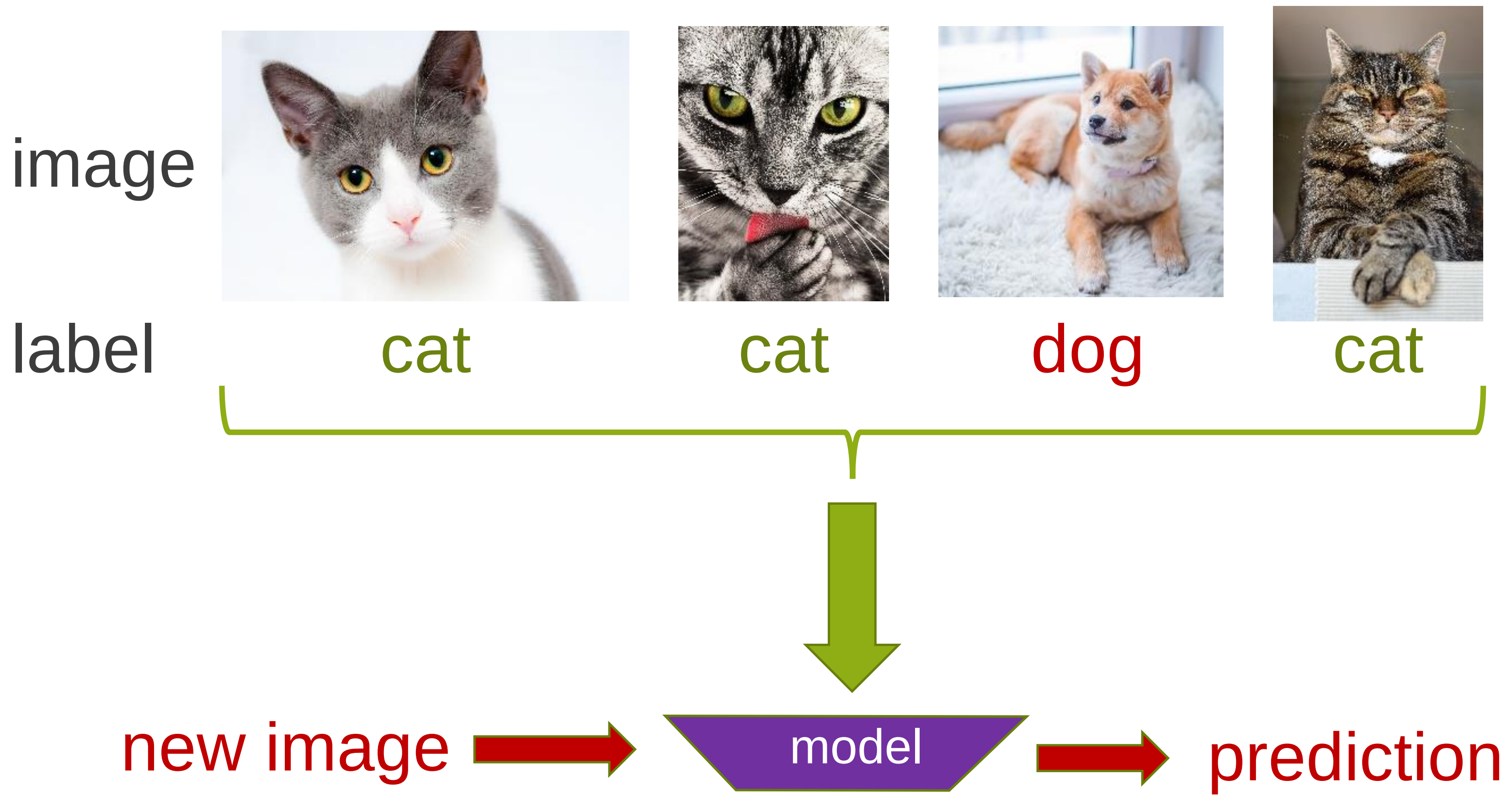
Program for machine vision

look for feature X
if this then that
else the other

look for feature Y

custom feature definition; difficult to write; complex, fragile

Machine Learning



All you need is lots of labeled data

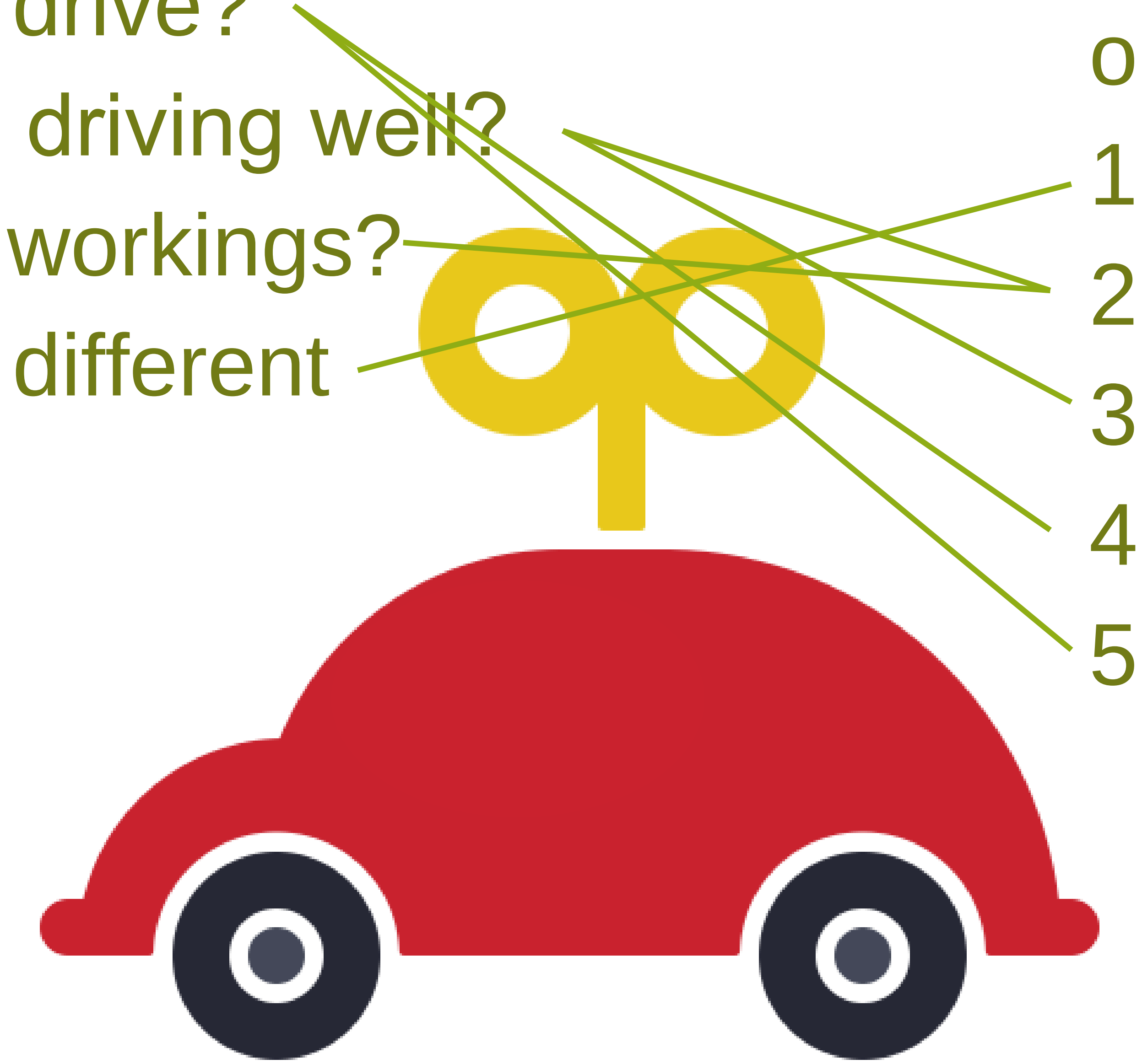
Direct control → Indirect control

Photos from Pexels

NO PEDALS, NO STEERING WHEEL: OK. NO WINDOWS: NOT OK!

Self-driving cars have raised some trust issues

- Why should I let it drive?
- How do I know it’s driving well?
- What are its inner workings?
- My driving style is different



Autonomous, closed loop operation absolutely requires:

1. Customization
2. Feedback
3. Man-machine cooperation
4. Building confidence
5. Tangible benefits

LESSONS TO LEARN

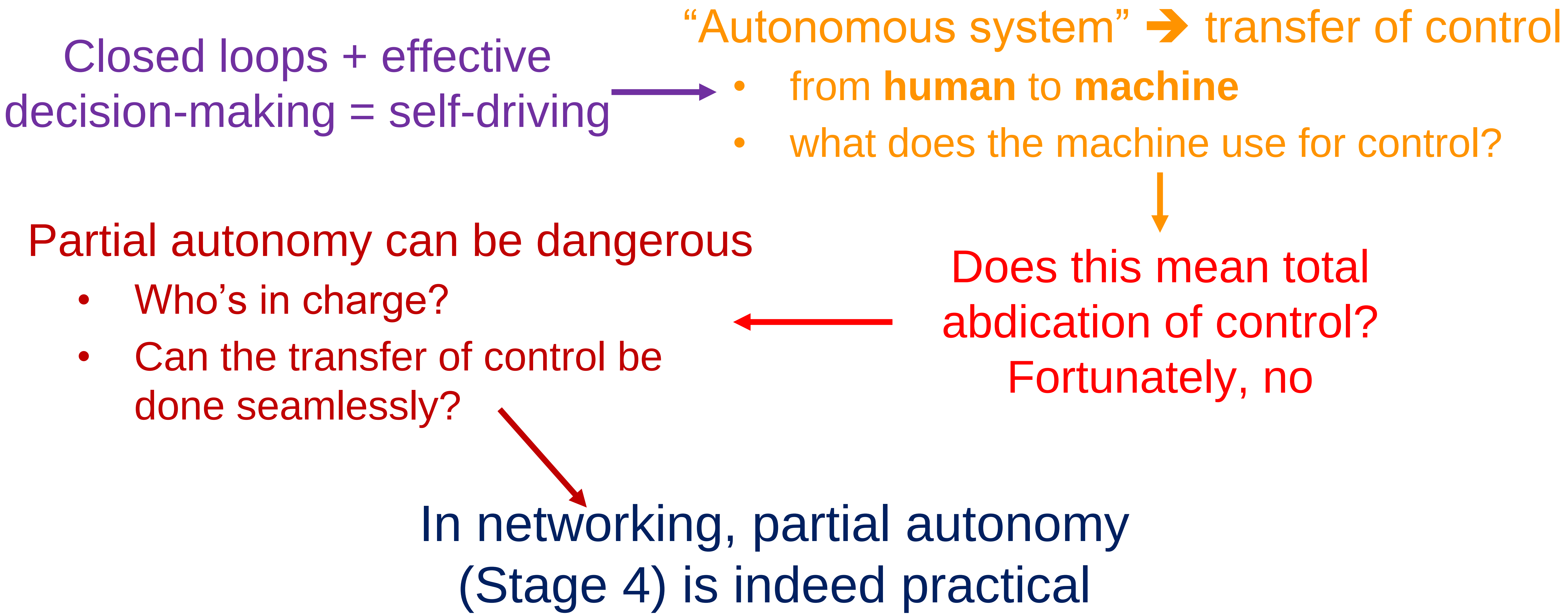
1. Give up control (but not *all* control)
2. ML is sexy, but may not be the place to start
3. Customization is crucial
4. Building trust is crucial

IMPLEMENTATION

Make it customizable, controllable, safe

Engineering Simplicity

ACHIEVING AUTONOMY



Closed loops + effective decision-making = self-driving

“Autonomous system” ➔ transfer of control

- from **human** to **machine**
- what does the machine use for control?

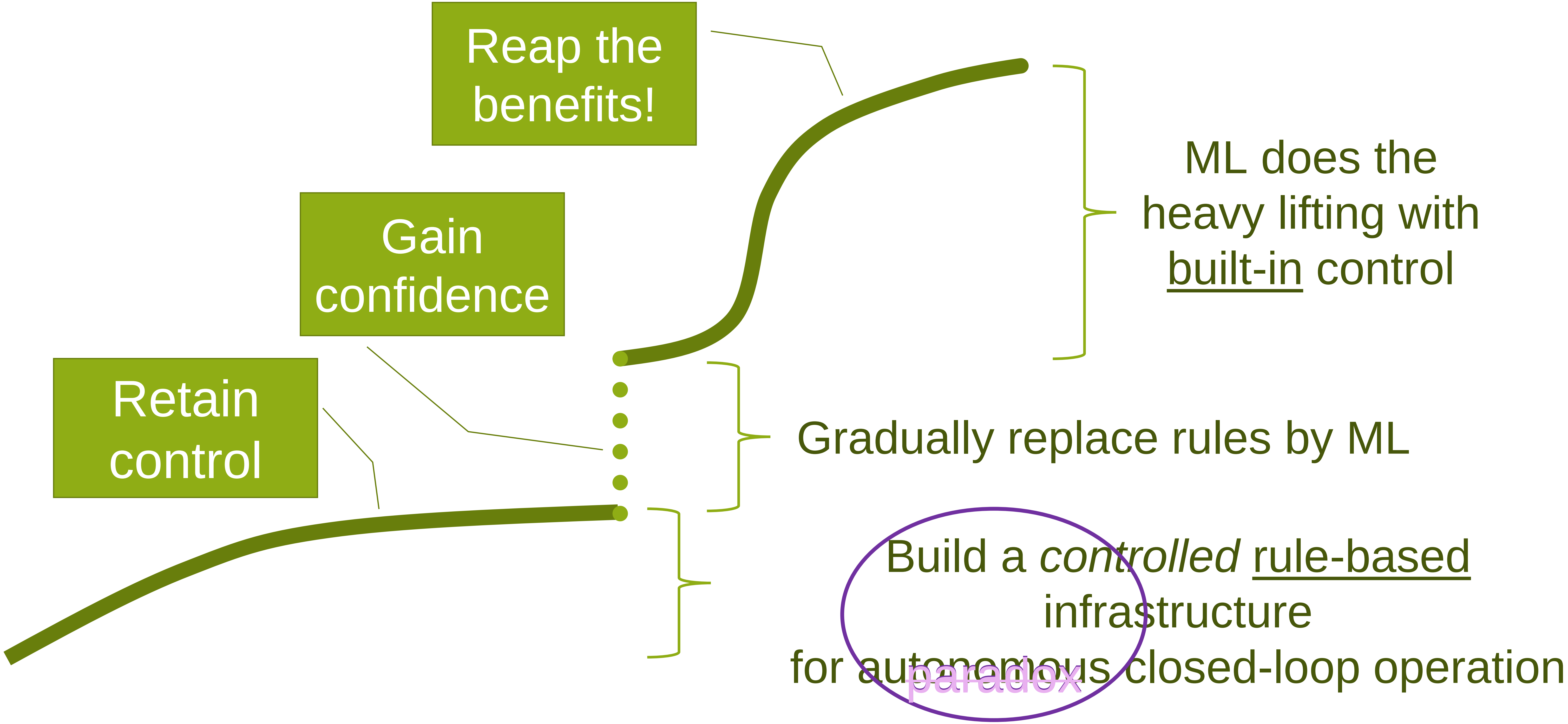
Does this mean total abdication of control? Fortunately, no

Partial autonomy can be dangerous

- Who’s in charge?
- Can the transfer of control be done seamlessly?

In networking, partial autonomy (Stage 4) is indeed practical

ROADMAP FOR DEPLOYING SELF-DRIVING NETWORKS



Reap the benefits!

Gain confidence

Retain control

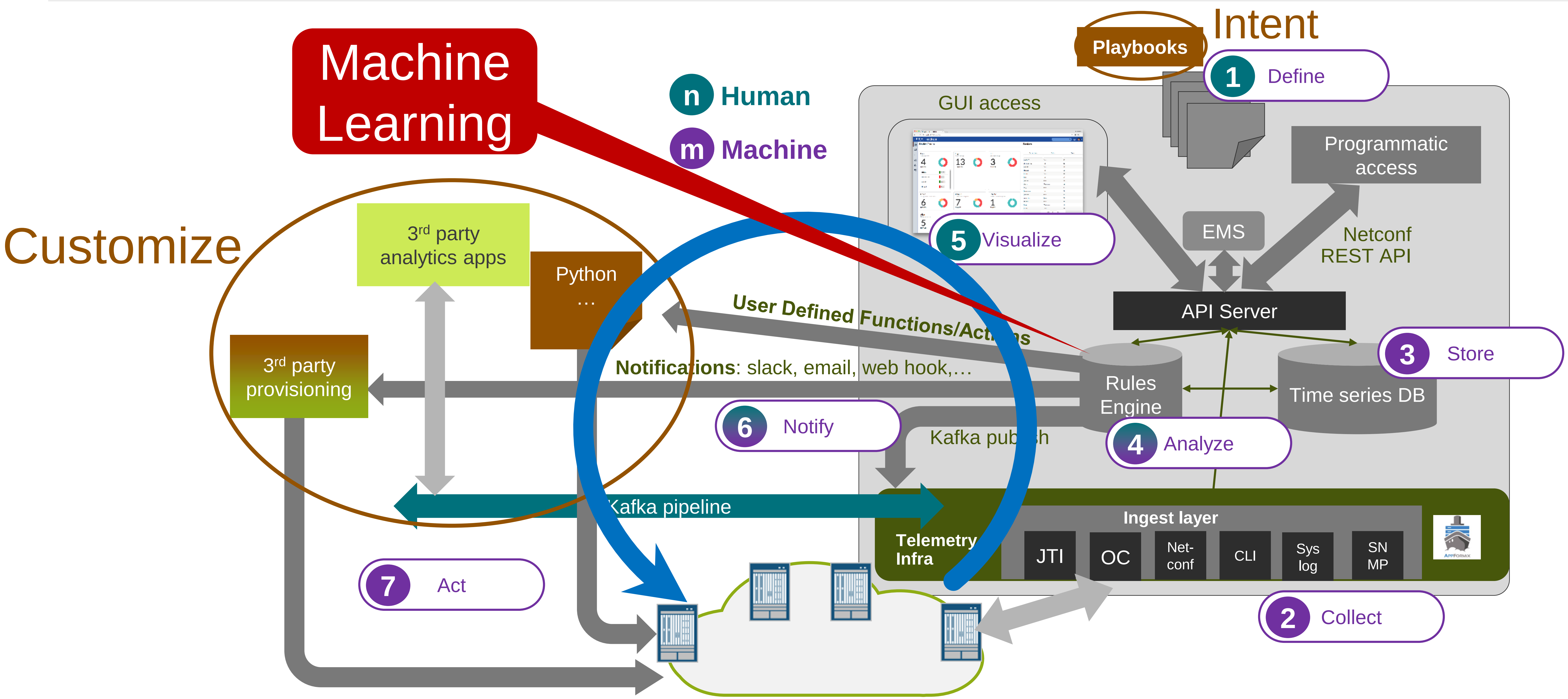
ML does the heavy lifting with built-in control

Gradually replace rules by ML

Build a *controlled* rule-based infrastructure
for autonomous closed-loop operation

paradox

ENABLING **CONTROLLABLE** *AUTONOMOUS* CLOSED LOOPS



GENERAL ARCHITECTURE FOR SELF-DRIVING

The same architecture can be used for various use cases:

- Device health
- Service health
- Underlay management
- …

The key is the playbook, written in a DSL

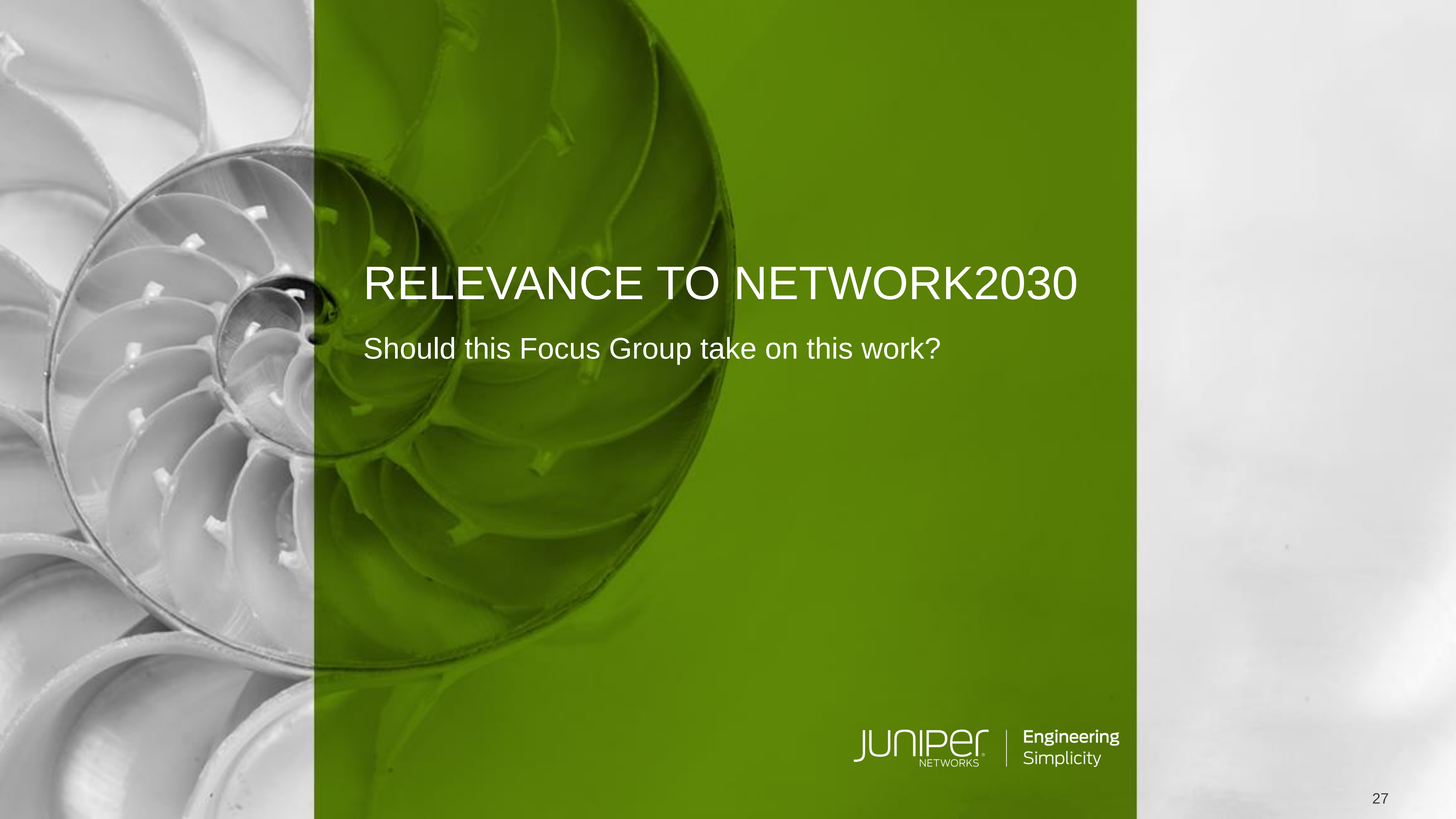
- Playbooks are the result of collaboration of engineers, support and network operators
- Playbooks are in a github repo, and can be customized by each network operator

(https://github.com/Juniper/healthbot-rules)

Things get really fun when playbooks interact

- For example, a “gray failure” of a link can lead to remapping the underlay
- Service (non-)health can be traced to an unhealthy device, which can then be fixed

This is where there is a need for more control until the interactions are better understood



RELEVANCE TO NETWORK2030

Should this Focus Group take on this work?

JUNIPER NETWORKS | Engineering Simplicity

BENEFITS

The goals of Self-Driving Networks are:

1. Capture human expertise & experience
2. Reduce mundane work (and thus errors)
3. Effectively utilize all possible data (telemetry, logs, …)
4. Improve efficiency on all fronts
5. Move to predictive operations (maintenance, service creation, …)

Human-Machine cooperation!

Many OPS groups face loss of talent

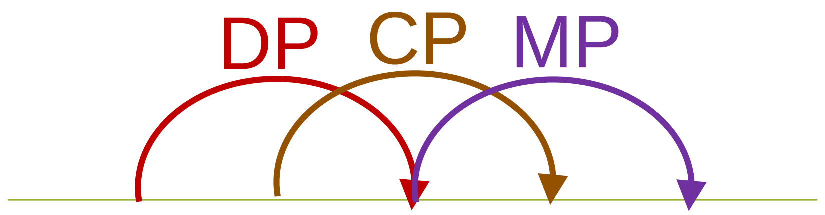
To free up time for innovation, for paying down “technical debt”, etc.

Humans can’t possibly process all the data available, nor prioritize it well

Better utilization of resources, dynamic shifting of workloads, …

This is the game-changer, but first, the infrastructure must be built

START WITH MANAGEMENT PLANE!



As networks get bigger
and more complex
(densification, IoT) …

As the data plane evolves
to meet new needs
(e.g., in-band telemetry) …

As SLAs get tougher
(e.g., latency → holographic)
…

As control plane actions have
more serious consequences
(e.g., remote surgery) …

Let’s bring new tools to bear on the
problem of managing networks!

SUMMARY

I’ve presented a new approach to managing networks, and a general architecture for **controllable** *autonomous* closed loops

- The starting point is rule-based, maintaining control via playbooks
  - Playbooks capture and distill human expertise and experience
  - This approach builds confidence and trust
- ***Closed loops can interact with each other, giving new use cases***
- Eventually, rules → ML, while retaining control and trust
  - Check out the playbooks on GitHub!

THANK YOU!