Open Problems in Robotic Anomaly Detection

Carnegie Mellon University
Software Engineering Institute

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

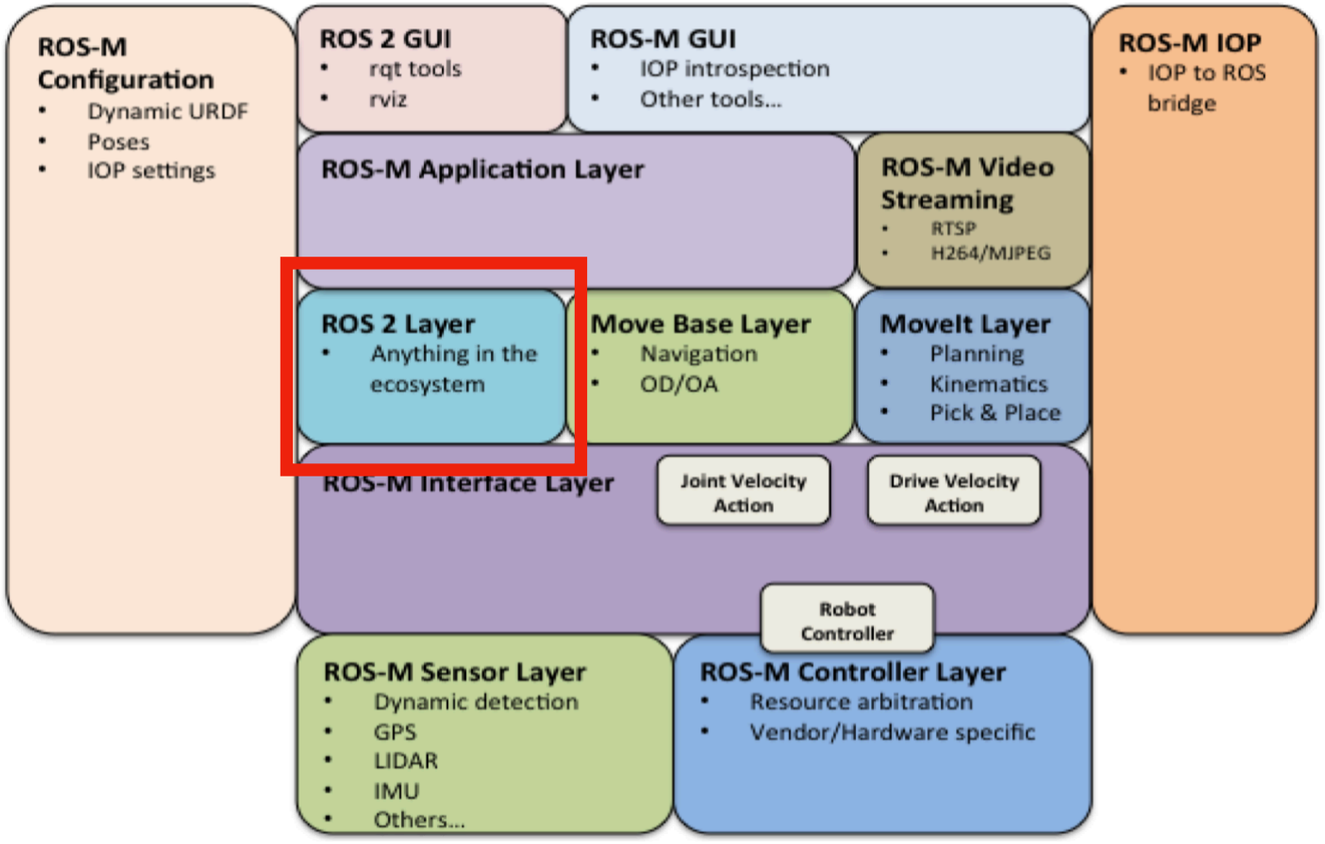
DM18-1157

Background

Photo credit: U.S. Army

Background

- Trusted and assured autonomy is the holy grail of unmanned robotics systems
- The future of Army robotics is built around ROS-M
- ROS-M is to be based on ROS 2
- It is unclear if ROS 2 can support the necessary anomaly detection tasks necessary for trusted and assured autonomy.

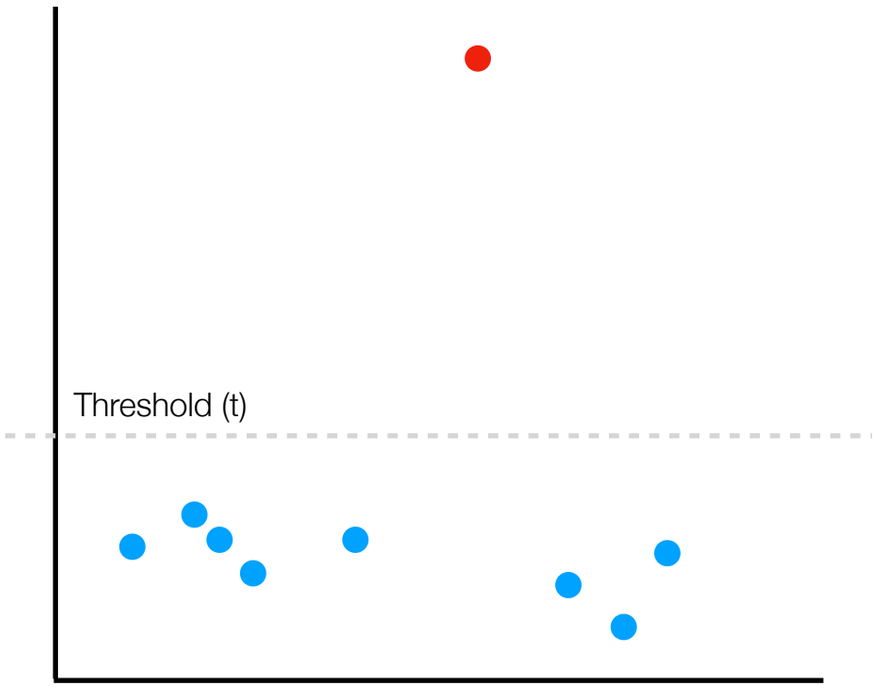


Background

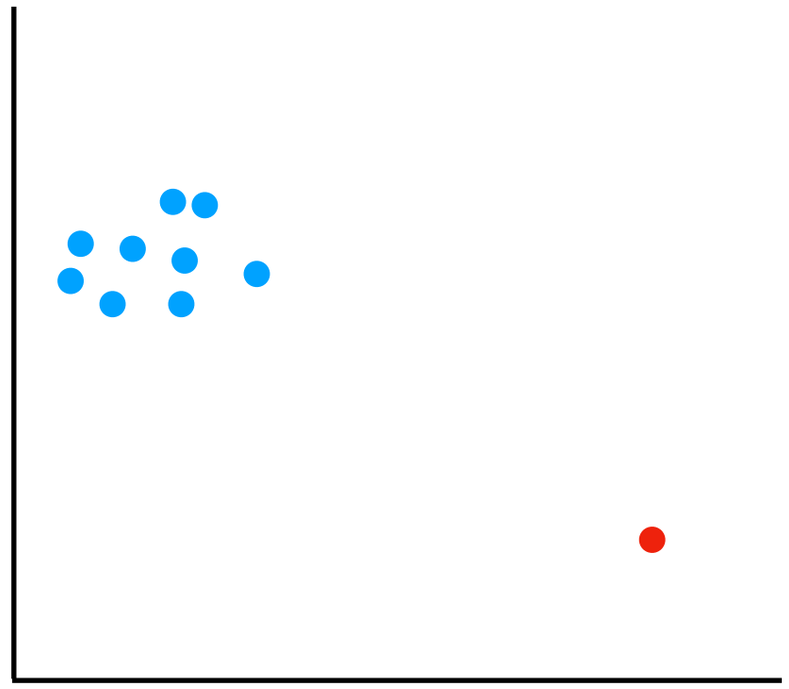
ROS-I (ROS Industrial) is a similar effort, but for industrial robotics

Photo credit: Carnegie Mellon University (CHIMP)

Anomaly Detection

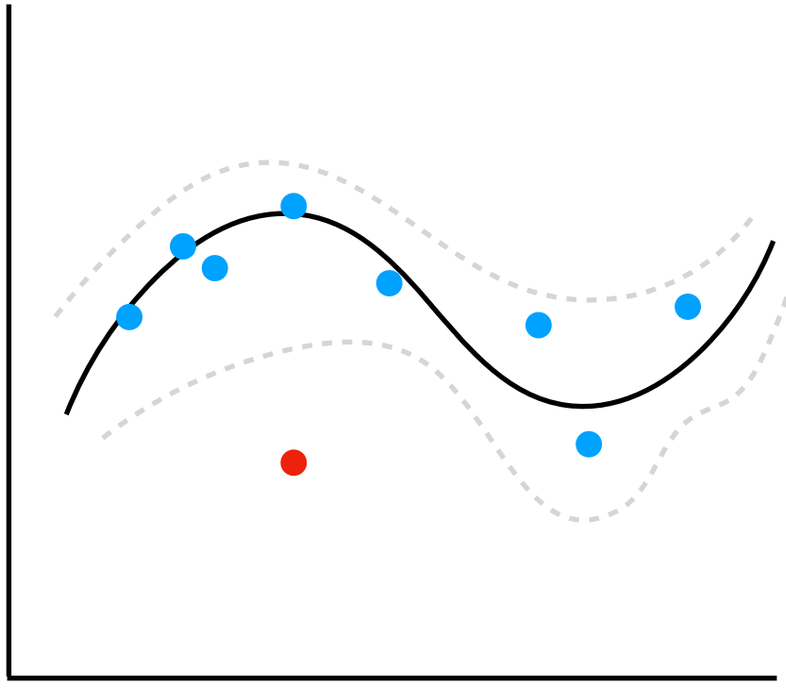


Extreme



Isolated

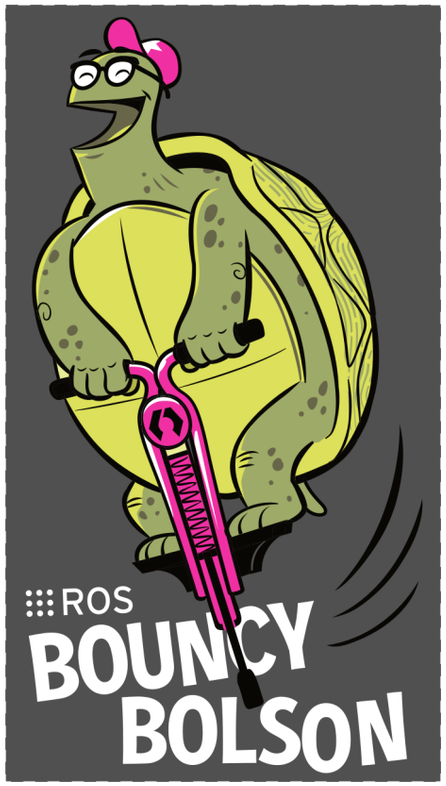
Anomaly Detection



Inconsistent with trusted model

Abstract

+

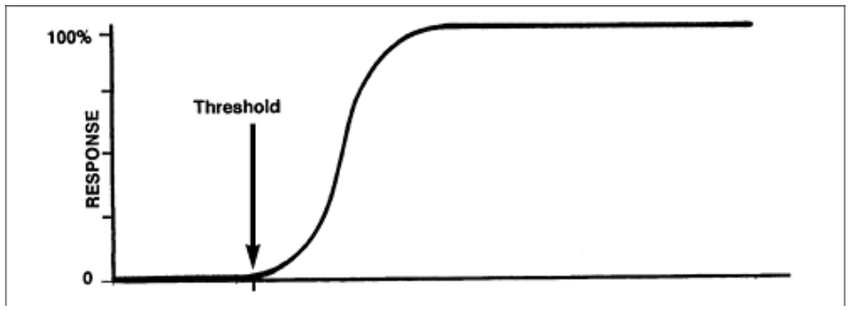


Non-malicious faults present many false alarms

Long-held belief that anomalies mean a failure of the system

but

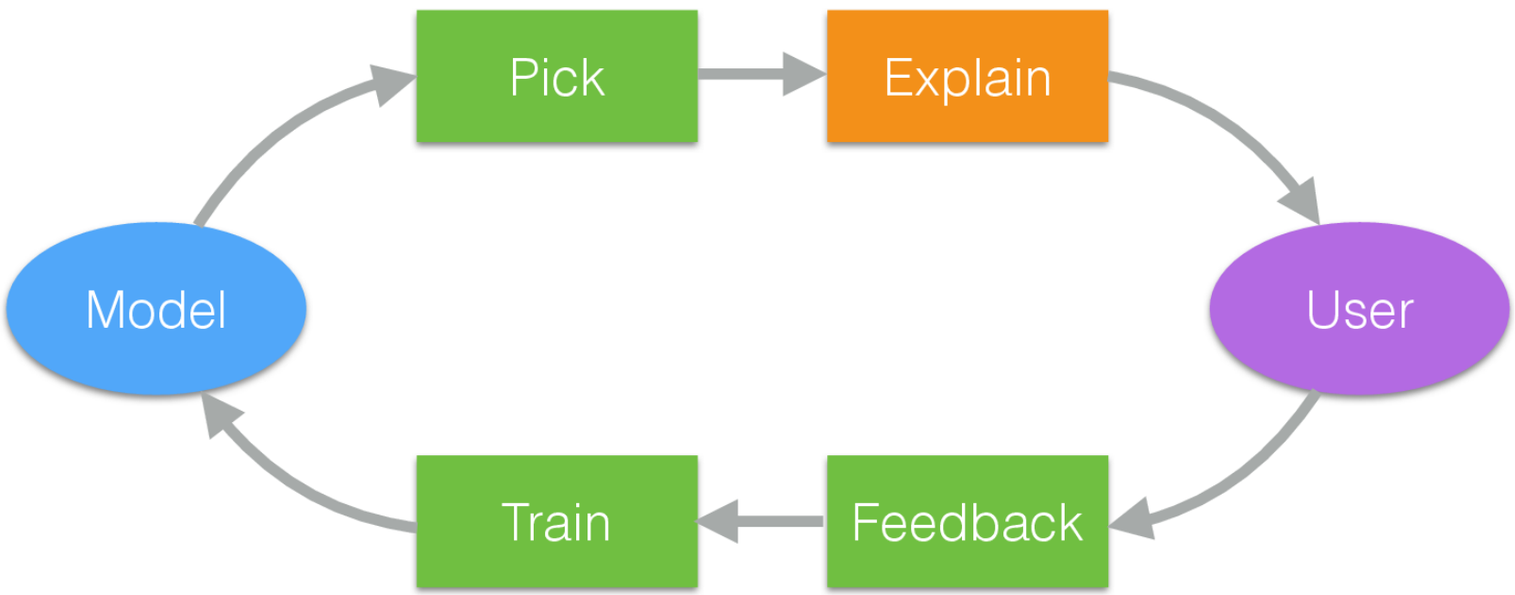
A robot could behave anomalously often without ever failing!



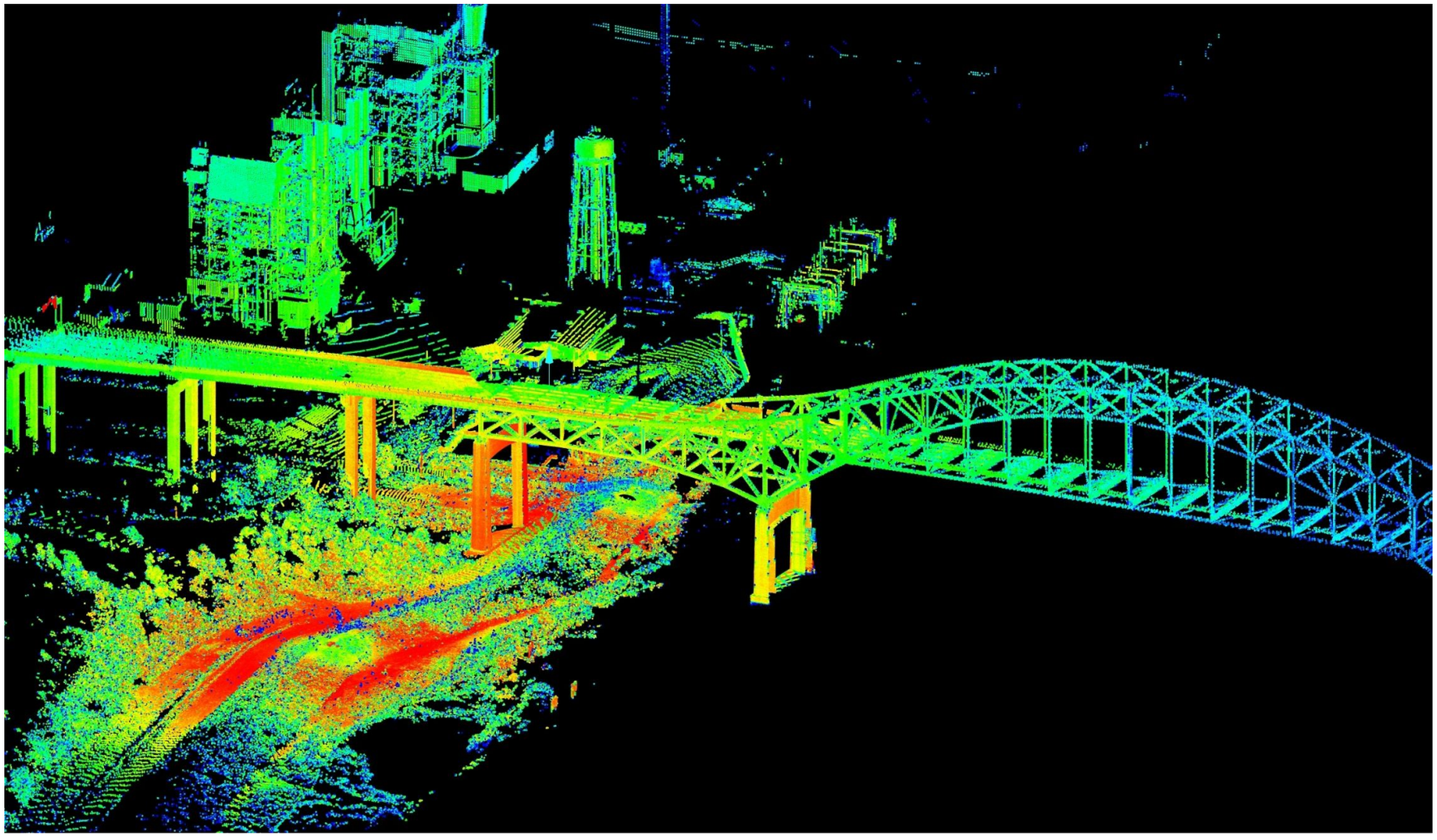
Non-malicious faults present many false alarms

- Threshold based
- Model based rejection
- Out-of-distribution, Bayesian analysis

Online human in the loop ML to learn from the operator



When is invalid data anomalous?

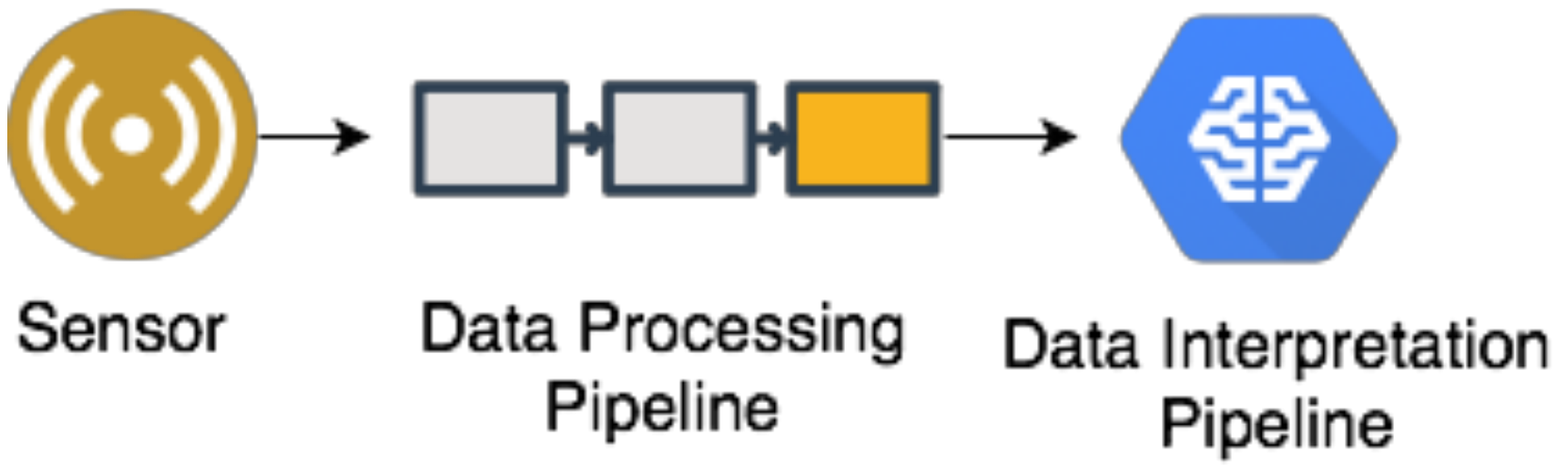


When is invalid data anomalous?

Data is never anomalous; interpretations are

Data can be flawed given a static interpretation framework

When is invalid data anomalous?



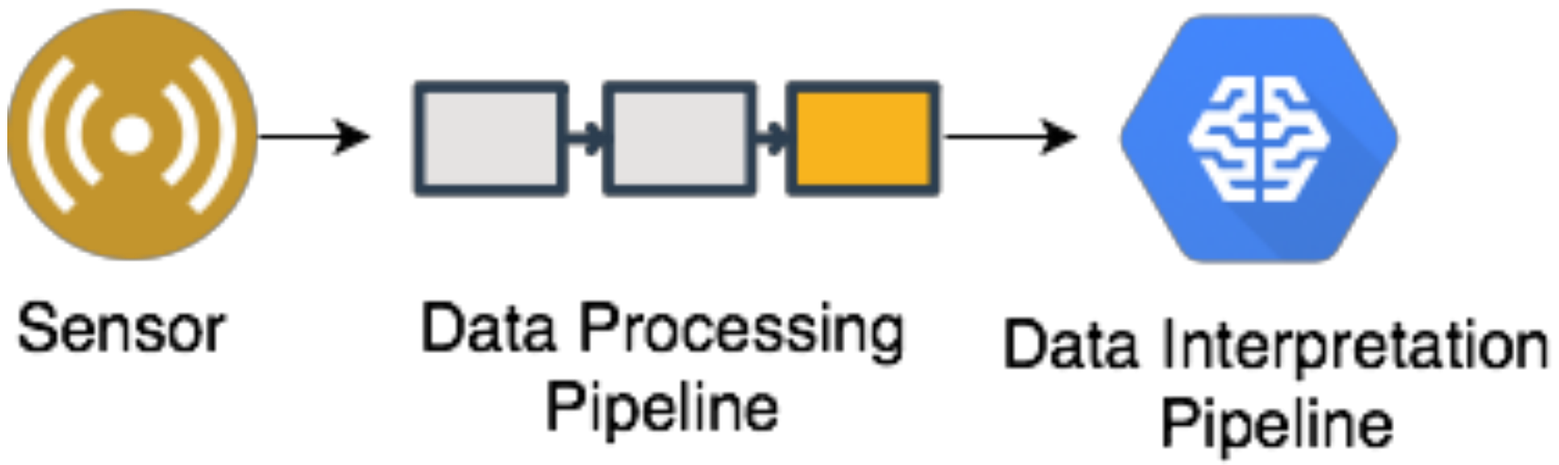
Invalid data $\not\propto$ Anomalous behavior

$$F(x \mid A)$$

Function that defines normal behavior

Set of assumptions

When is invalid data anomalous?



Invalid data $\not\propto$ Anomalous behavior

$$F(x \mid A)$$

Function that defines normal behavior

Set of assumptions

How do we update our assumptions?

We could avoid them altogether?

- Non-parametric methods let us do this

Model the assumptions and condition our anomaly detection algorithm on the assumption model?

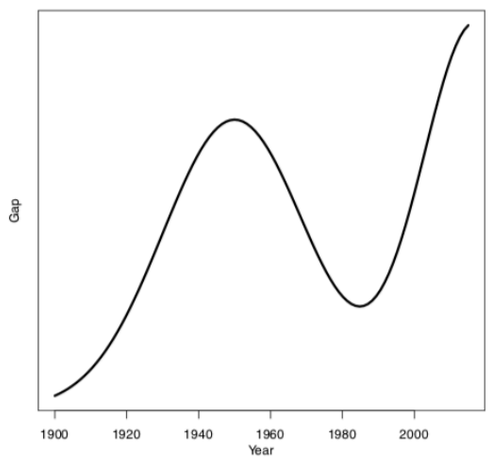
Other approaches?

The Role of Assumptions in Machine Learning and Statistics: Don't Drink the Koolaid!

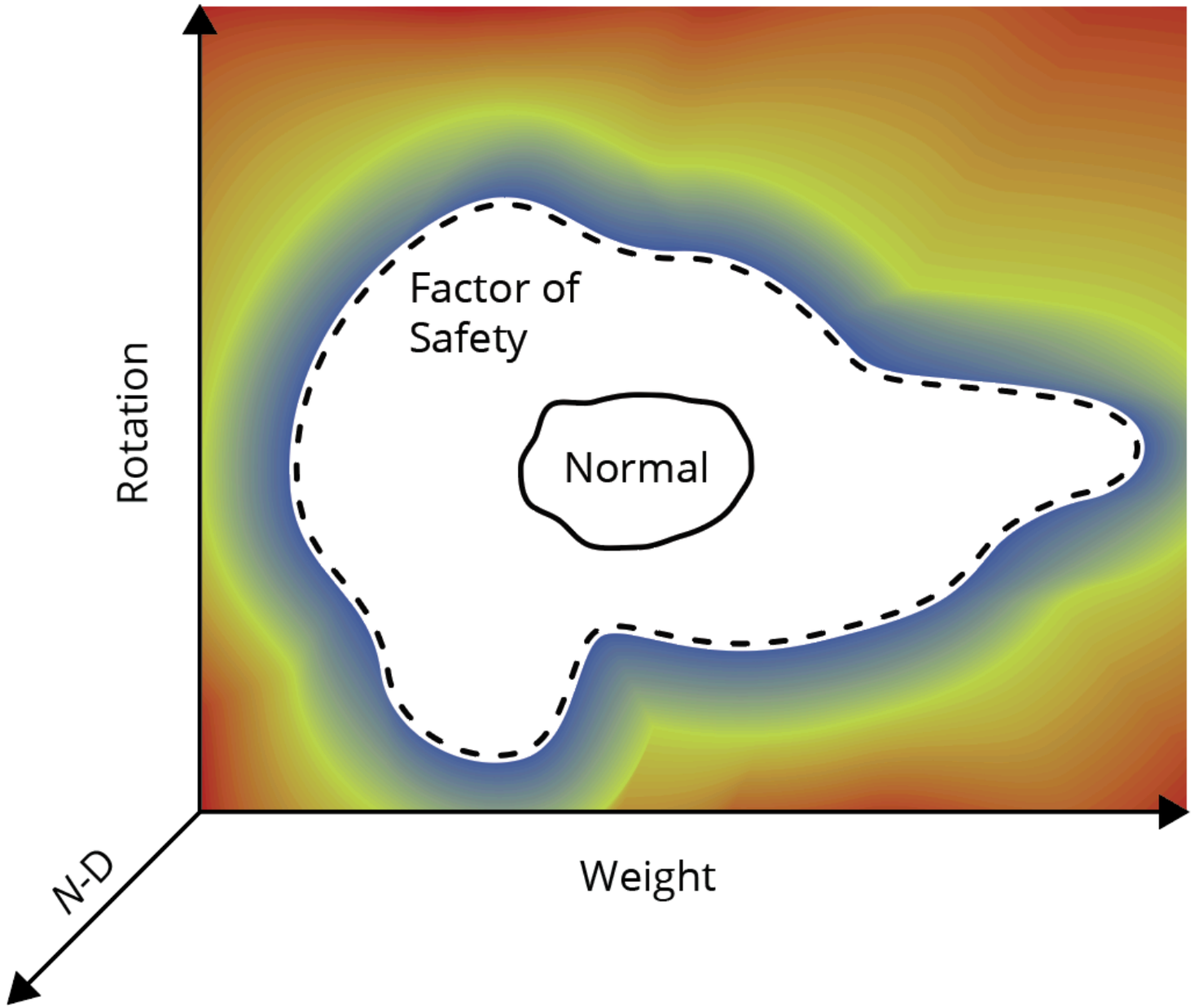
Larry Wasserman
April 12 2015

1 Introduction

There is a gap between the assumptions we make to prove that our methods work and the assumptions that are realistic in practice. This has always been the case, and the size of the gap varies with time. But, due to the ubiquity of high dimensional problems, the gap has become dangerously wide. It looks like this:



Intentional anomalous behavior and emergency stops



Given some state $\phi \in OC$, when does it represent anomalous behavior?

Hierarchies of systems with shared functionality

A robot is defined as:

- a collection of k nodes $V = \{v_1 , ..., v_k \}$, where some nodes are connected by directed edges $E = \{(v_i , v_j)\}$ variously representing physical anchoring, energy flow, or information flow of various kinds,
- the graph is defined as $G = (V, E)$,
- nodes can be grouped in the form of $\{ v_x \mid f(v_x) \} \exists_v x \in C$, where $f(x)$ represents a predicate function that returns true if v_x has a certain functionality, and C represents the overall set of all groups in the robotic system,
- and v_x is a member of only one subset of C

Hierarchies of systems with shared functionality

Composability!

Behavior of nodes V:

$B = [b_1 , ..., b_k]$, where $|B| = |V|$.

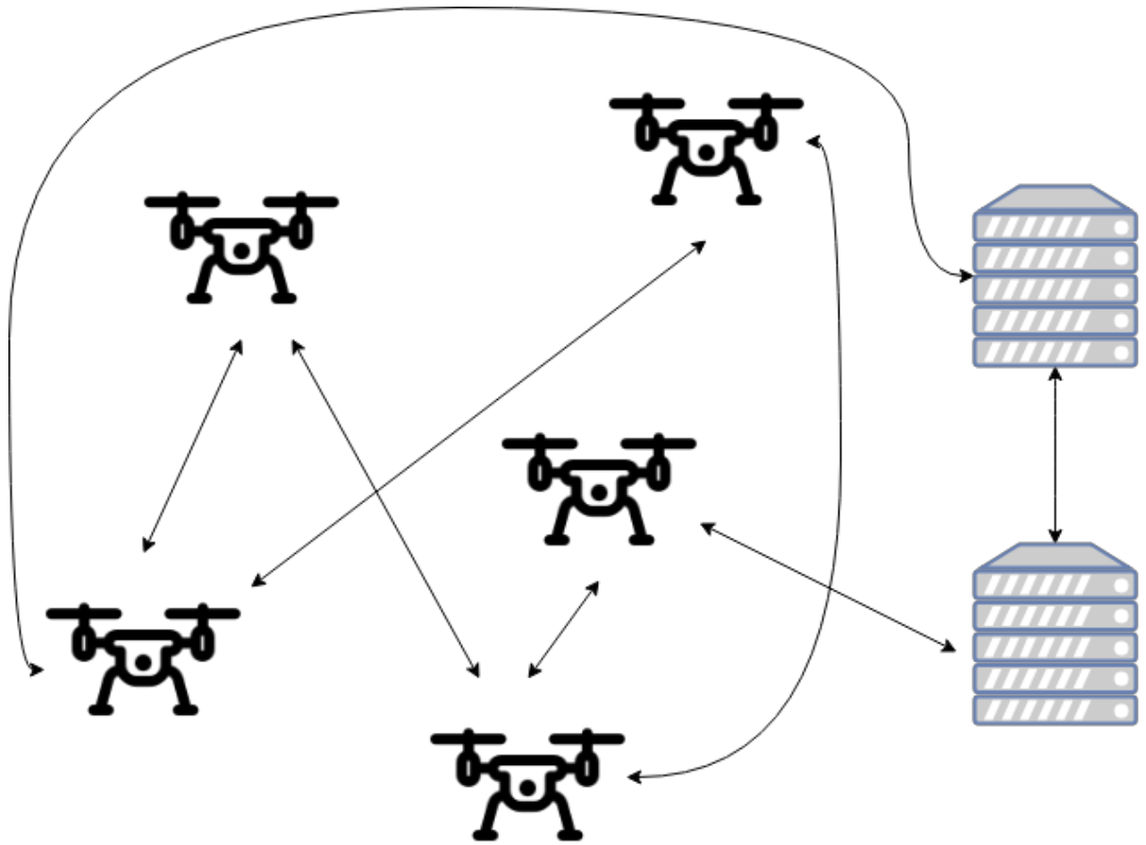
Vector of constants $\Phi = [\alpha_1 , ..., \alpha_k]$, $|\Phi| = |B|$.

Linear composability is then defined by:

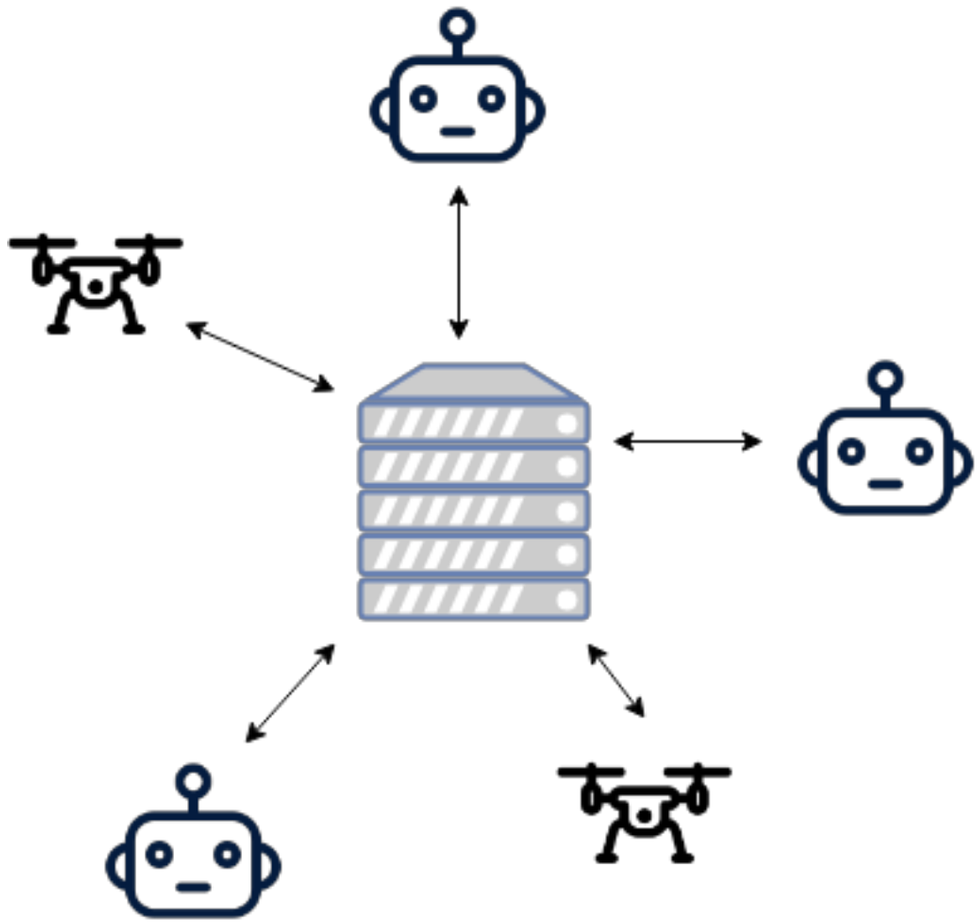
$$\Phi^T \cdot B = a_1 b_1 + ... + a_k b_k$$

This entire relationship is decomposable!

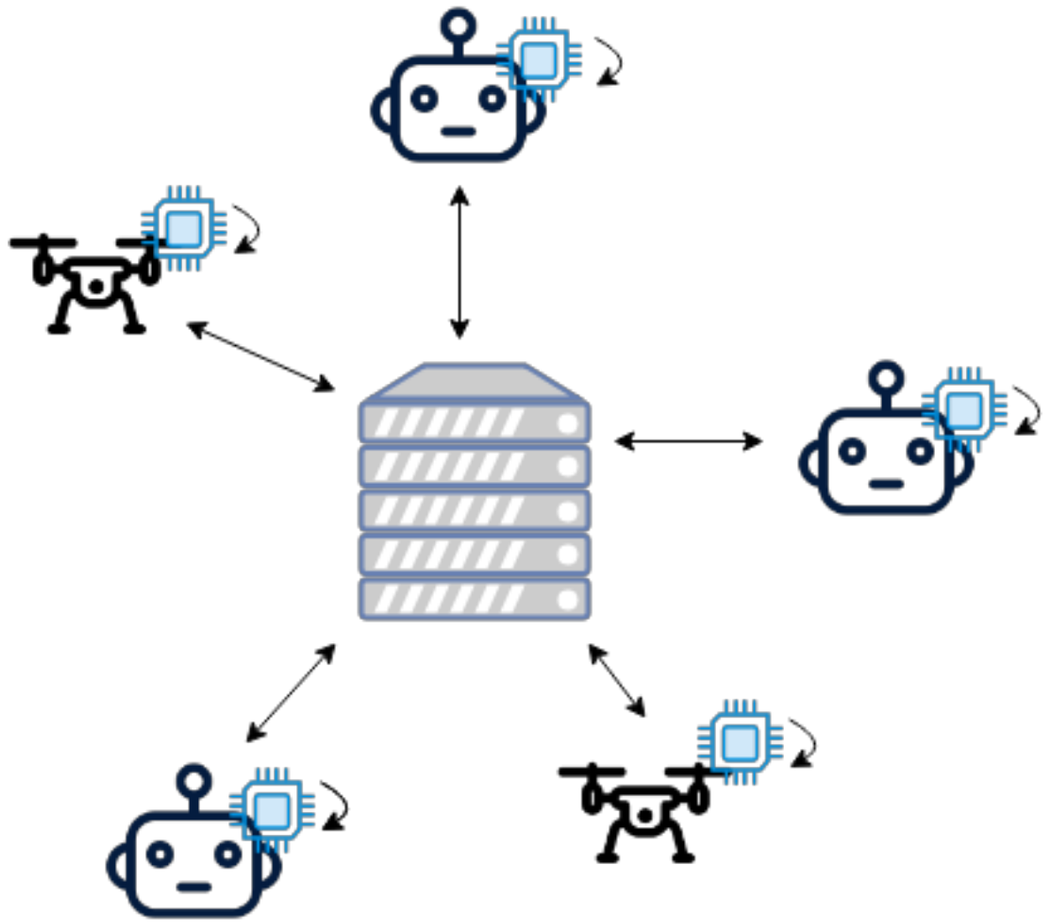
Distribution of computation across hosts



Distribution of computation across hosts



Distribution of computation across hosts



Distributed and Efficient ML

Parallelized Stochastic Gradient Descent

Martin A. Zinkevich
Yahoo! Labs
Sunnyvale, CA 94089
maz@yahoo-inc.com

Markus Weimer
Yahoo! Labs
Sunnyvale, CA 94089
weimer@yahoo-inc.com

Alex Smola
Yahoo! Labs
Sunnyvale, CA 94089
smola@yahoo-inc.com

Lihong Li
Yahoo! Labs
Sunnyvale, CA 94089
lihong@yahoo-inc.com

HOROVOD

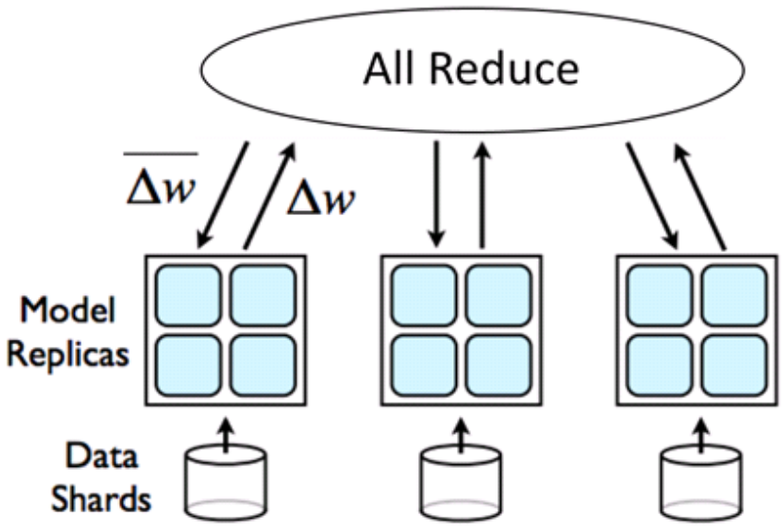


Photo credit: Apache Software Foundation

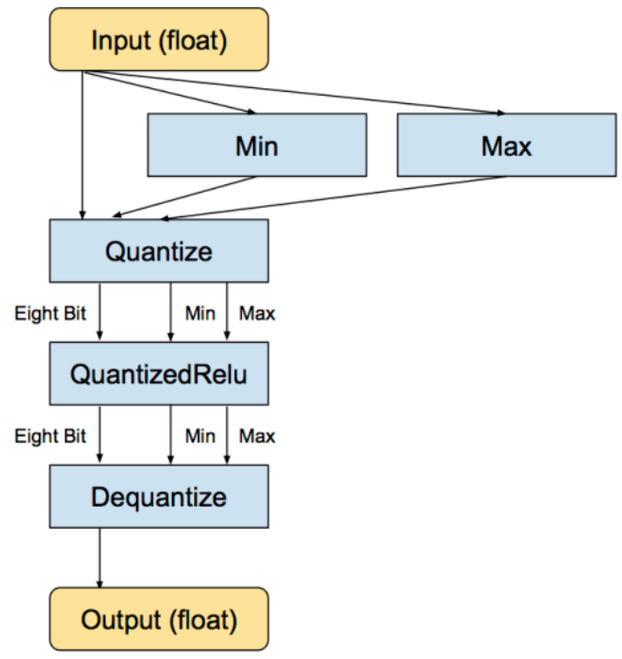
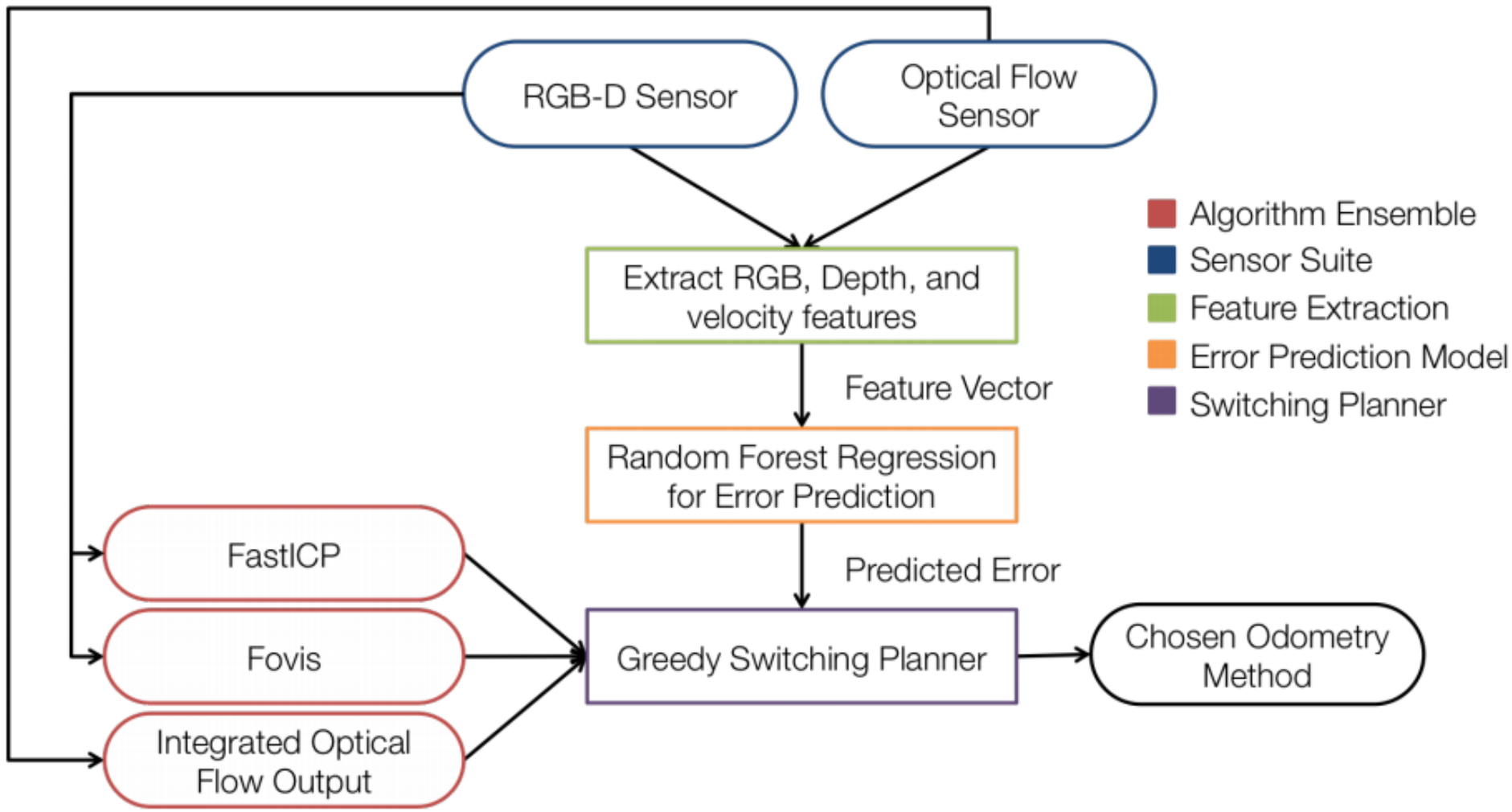


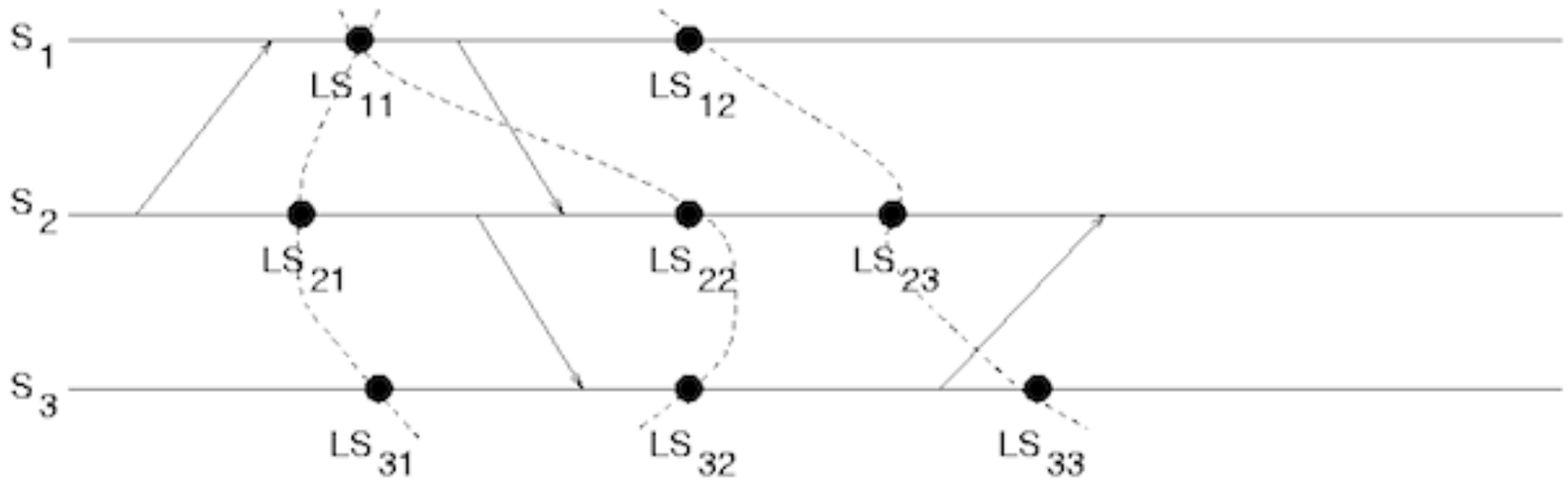
Photo credit: Pete Warden

Fixing anomalies on the fly

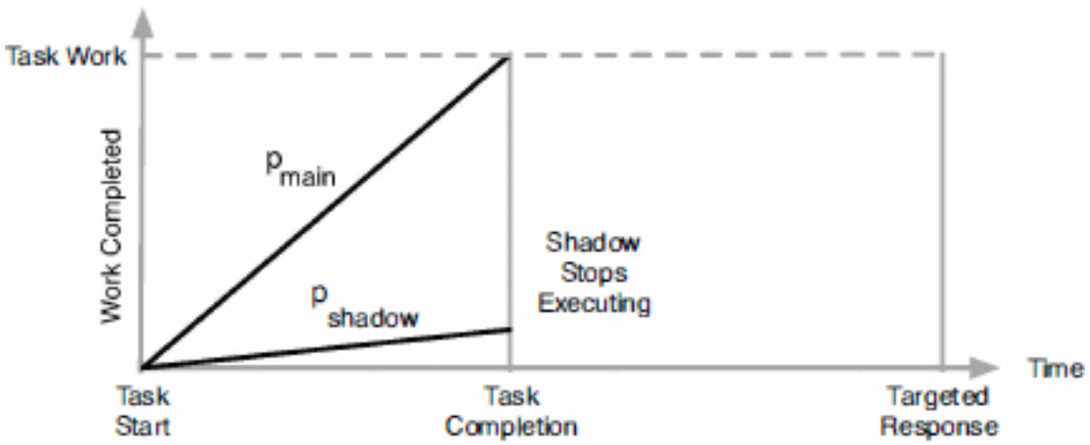


Kristen Holtz, Daniel Maturana, and Sebastian Scherer. "Learning a Context-Dependent Switching Strategy for Robust Visual Odometry."

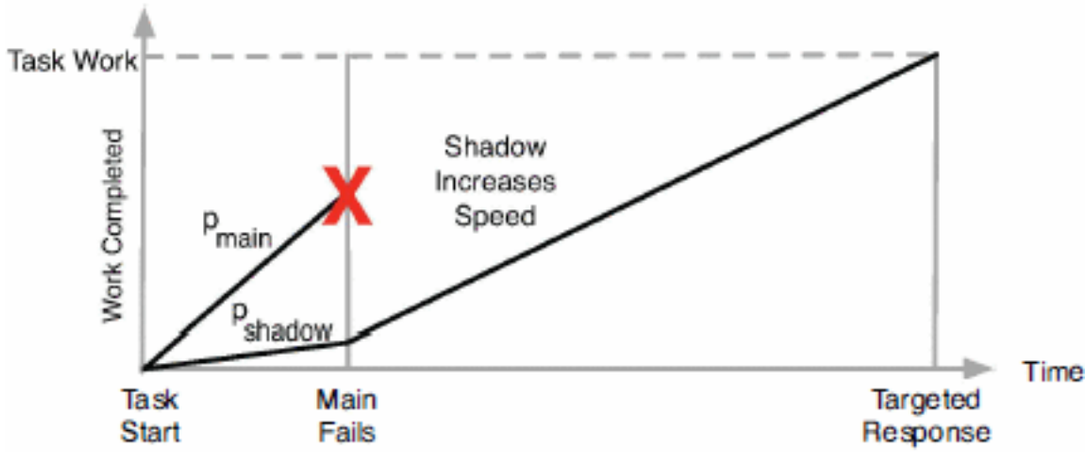
Fixing anomalies on the fly



Fixing anomalies on the fly



(a) Case of no failure



(b) Case of failure

B. Mills, T. Znati, and R. Melhem. "Shadow Computing: An energy-aware fault tolerant computing model."

Open Problems in Robotic Anomaly Detection

Ritwik Gupta[1], Zachary T. Kurtz[1], Sebastian Scherer[2], and Jonathon M. Smereka[3]

***Abstract*— Failures in robotics can have disastrous consequences that worsen rapidly over time. This, the ability to rely on robotic systems, depends on our ability to monitor them and intercede when necessary, manually or autonomously. Prior work in this area surveys intrusion detection and security challenges in robotics, but a discussion of the more general anomaly detection problems is lacking. As such, we provide a brief insight-focused discussion and frameworks of thought on some compelling open problems with anomaly detection in robotic systems. Namely, we discuss non-malicious faults, invalid data, intentional anomalous behavior, hierarchical anomaly detection, distribution of computation, and anomaly correction on the fly. We demonstrate the need for additional work in these areas by providing a case study which examines the limitations of implementing a basic anomaly detection (AD) system in the Robot Operating System (ROS) 2 middleware. Showing that if even supporting a basic system is a significant hurdle, the path to more complex and advanced AD systems is even more problematic. We discuss these ROS 2 platform limitations to support solutions in robotic anomaly detection and provide recommendations to address the issues discovered.**

I. Introduction

Anomaly detection (AD) is an increasingly important area of study in the field of robotics as robotic systems tend towards higher levels of autonomy. Being able to predict, identify, and correct these anomalies is critical, especially when the robotic systems can have a direct or indirect impact on human life. Unfortunately, while all versions of anomaly detection seek to identify things that are anomalous, there is still considerable variation in precisely what this means:

1) Extreme: The point lies above a threshold t.
2) Isolated: In some metric space, the distance to other points is greater than t except for at most n of other very nearby points (a point at the center of a highly bimodal distribution can be isolated and not extreme).
3) Abnormal (or *inconsistent with a trusted model*): As an example, an auditor keeps track of the ratio of total income to total taxes paid for a collection of organizations. One organization is far larger than the others, with income and taxes being both extremely high. However, the ratio of taxes to income for this large organization is comparable to the ratio for smaller organizations, and the auditor considers it normal. Thus, a point can be both extreme and/or isolated and yet still fail to be abnormal.

The differences between the senses above are conceptually superficial. For any space containing an isolated point, there exists a simple transformation of the space that results in the isolated point becoming an extreme value. Similarly, the size (in terms of income and taxes) of an organization is really a distraction if the ratio of income to taxes is what matters, so why not just talk about that ratio? Unfortunately, while these kinds of conceptual connections between competing notions of anomalousness are trivial for simple examples, they become less trivial as the dimension of the space grows.

The anomaly detection task is especially challenging when we are asked to treat the data as a black box, with no *a priori* insight into what is "normal". A general-purpose anomaly detection algorithm will require considerable sophistication to automatically notice the relationship between income and taxes without any prior knowledge of finance. Accordingly, varying techniques of anomaly detection in robotic monitoring focus on predefined relationships of what is a "normal range" of operation [1], [2], [3], [4], [5], however, as we show in this work, there are still several open problems in robotic anomaly detection that significantly degrade the assumption of being able to define that "normal range".

Finally, we demonstrate the need for additional work in these areas by providing a case study which examines the limitations of implementing a basic anomaly detection (AD) system in the Robot Operating System (ROS) 2 middleware [6], which is an attempt to revise and improve many engineering decisions from the ROS 1 platform [7]. ROS has often been difficult to work with and requires specific engineering guidelines which are not conducive to real-time anomaly detection. Accordingly, we draw the conclusion that if even supporting a basic system is a significant hurdle, the path to more complex and advanced AD systems is even more problematic. We discuss these ROS 2 platform limitations to support solutions in robotic anomaly detection and provide recommendations to address the issues discovered.

II. Open problems with regards to robotic AD systems

A. Non-malicious faults present many false alarms.

False positives and false negatives have been well studied in AD and intrusion detection systems [8], [9], [10]. It is a long-held belief that an anomaly means a failure of a system directly. However, *not all anomalies represent failures*. A robot can behave anomalously frequently without ever failing, resulting in a large amount of false alarms that,

*This work was sponsored by the U.S. Army Tank Automotive Research, Development, and Engineering Center (TARDEC)

[1]Ritwik Gupta and Zachary T. Kurtz are with the Software Engineering Institute, Carnegie Mellon University, 4500 Fifth Avenue, Pittsburgh, PA 15213, USA {rgupta, ztkurtz}@sei.cmu.edu

[2]Sebastian Scherer is with the Robotics Institute, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh 15213, PA, USA basti@andrew.cmu.edu

[3]Jonathon M. Smereka is with U.S. Army TARDEC, Warren, MI 48397, USA jonathon.m.smereka.civ@mail.mil