

hackHub | Trojan Horse

Problem Statement

Fault tolerance and detection is increasingly important for robots, especially those in remote or hazardous environments. Robots need the ability to effectively detect and tolerate internal failures in order to continue performing their tasks without the need for immediate human intervention. Recently, there has been a surge of interest in robot fault tolerance, and the subject has been investigated from a number of points of view.

Abstract

We aim to create a cloud based system which aims to create a feedback system which takes values from control systems placed far away from each other, and with minimal human intervention, moderates the power supplied to industrial machinery and at the same time if any of the received data shows abnormal variations or shows sign of some potential mishap, predict any such disasters and notify concerned personnel immediately.

Specifications

- One Master server with 'n' raspberry pi / smart devices
- Scalable to a limit depending on demand
- Communication rate at 9600 bps (can be increased if instantaneous readings are required)
- Full duplex communication between master and slaves
- Easy integration with amazon alexa to handle erroneous situations instantly
- Mobile App Support through amazon alexa with custom skills
- Custom cloud based server platform
- On-board sensor-based fault prevention in case of extreme situations to protect the IoT devices
 - No human intervention required
 - Sensors like
 - Temperature | Monitors real time heating and trips to a cooling circuit
 - Current | In case of fluctuation or high current spike, trips to protection mode

- Hardware based fault prevention - Protects circuits from potential harm
- Admin access to visualise status of every sensor and raspberry/arduino in an interactive and dynamic UI
- Custom Server based processing of data in real time
- Automatic pinpoint restart of devices in case of failures

Methodology

- Reading Sensor Data
 - Using a microcontroller , all sensor values are recorded and using Serial Peripheral Interface(SPI) is sent to the microprocessor (PI)
 - Our sensors include Hall current sensor which detects change in current throughput through a particular device
 - Temperature sensor which works in an environment independent from the rest of our system which enables it to function even during system failures
- Parsing and Uploading Sensor Data
 - All sensor data being received is parsed and sent to the cloud to process in real time
 - The data is updated on ThingSpeak Servers | For cost cutting and faster performance, custom server based database can be implemented
- Processing Incoming Data and Taking Actions
 - Server processes incoming data and provides visualisation of sensor values over time
 - Abnormal Sensor data is immediately identified and appropriate actions are taken
 - Faults occurred are immediately displayed on the UI
 - Faults such as high current throughout are automatically addressed by circuit switching or power management
 - Option for user intervention by integration of Amazon Alexa
 - Custom skills added on Amazon Web Services to handle fault recovery
 - Server can handle multiple incoming requests based on severity of problem
 - Server provides capabilities for autonomous or/and manual intervention during state of distress

FAULT PREVENTION:

Instead of using resources for fault rectification, any such system's main priority should be prevention of the faults in first place.

Our system continuously monitors the data and analyses whether the current trend shows any deviation from the nominal configuration of the device.

For example, if some servo motor, which is a part of an industrial robotic arm, starts behaving erratically

Requirements

To develop a system that can:

1. Smartly detect fault
2. Restart faulty device
3. Implement fault prevention
4. Fault avoidance
5. Fault detection
6. Fault recovery strategies

Plan of Action

- Making a IOT network with pi and arduino and multiple sensors which constantly send value to pi
 - Interface arduino with sensors
 - Interface a servo motor, dc motor, stepper with Arduino and temp module.
 - Send data serially to pi
 - Upload data to server
- Interface for user
 - Status of multiple devices
 - Sensor Values being recieved- real time monitoring of sensors
 - Mapping of devices present in the network
 - Option for scalability
 - Show added devices
 - Show status if a device is removed